onwards

Hello, 2021

Hello, 2021

Hello, 2021. If anyone has a 747 in need of a paint job this is something I would like to see happen "**for reals** `https://en.wiktionary.org/wiki/for_reals` ". I mean it.

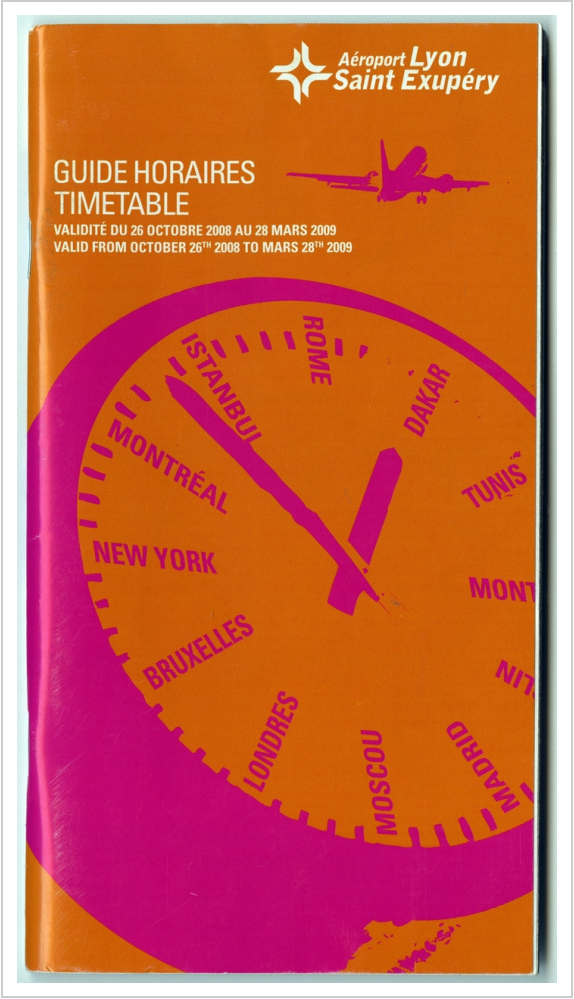
c'est la
danse de
demain

2021-01-05

things I have written about elsewhere #20210114

Tools for Complex and Ambiguous Dates at SFO...

Tools for Complex and Ambiguous Dates at SFO Museum



timetable: Lyon - Saint Exupéry Airport. Paper, ink. Gift of the William Hough Collection, SFO Museum Collection.
2009.108.042

This was ***originally published on the SFO Museum Mills Field***

weblog https://millsfield.sfomuseum.org/blog/2021/01/14/edtf/ *, in January 2021.*

Today, we are pleased to announce the release of four tools for working with **Extended DateTime Format** https://www.loc.gov/standards/datetime (EDTF) dates. Three of the four tools are general-purpose utilities that build on one another. The fourth is an SFO Museum-specific application of these tools and is being shared in the spirit of generosity and as a reference implementation for how these tools might be adapted for other use-cases.

EDTF is a standard for representing complex, and sometimes ambiguous, dates developed under the umbrella of the **Library of Congress** https://www.loc.gov/standards/datetime/background.html and describes itself this way:

EDTF defines features to be supported in a date/time string, features considered useful for a wide variety of applications

Date and time formats are specified in ISO 8601, the International Standard for the representation of dates and times. ISO 8601-2004 provided basic date and time formats; these were not sufficiently expressive to support various semantic qualifiers and concepts than many applications find useful. For example, although it could express the concept "the year 1984", it could not express "approximately the year 1984", or "we think the year is 1984 but we're not certain". These as well as various other concepts had therefore often been represented using ad hoc conventions; EDTF provides a standard syntax for their representation.

Further, 8601 is a complex specification describing a large number of date/time formats, in many cases providing multiple options for a given format. Thus a second aim of EDTF is to restrict the supported formats to a smaller set.

EDTF functionality has now been integrated into **ISO 8601-1:2019** `https://www.iso.org/standard/70907.html` and **ISO 8601-2:2019** `https://www.iso.org/standard/70908.html`, the latest revisions of ISO 8601, published in March 2019.

go-edtf

A pioneer in avionics, United was the first transportation company to install a two-way, coast-to-coast teletype service. Set up in 1938, it linked stations all along the Main Line from New York to San Francisco. A radio relay system was also used to track airplane movement. As an airplane passed radio receivers located along each route segment, a signal was automatically transmitted to the ground station's flight dispatch clock. The dispatcher would then move a pin with the corresponding flight number one position closer to the center of the clock board to plot the flight's progress. See "Flying the Main Line: A History of United Air Lines", on display, post-security, in Terminal 3. - ***Instagram, September 19, 2017***

The first tool we've released is a **Go language** `https://golang.org` package for validating, and parsing EDTF date strings. It implements all three "compliance" levels defined in the **2019 EDTF specification** `http://www.loc.gov/standards/datetime/` and includes a comprehensive test suite.

It also defines a custom data structure called an `EDTFDate` to parse

and expose the semantics encoded in a date string. An `EDTFDate` is comprised of two "date spans" each of which contains a "date range" containing an upper and lower date. For example, the EDTF string "1950-06-XX/2004-07-03" which denotes a date "sometime between June, 1950 and July 03, 2004" is represented with the following date ranges and date spans:

EDTF component	Span	Range	DateTime	Timestamp
1950-06-XX	Start	Lower	1950-06-01T00:00:00Z	-618105600
1950-06-XX	Start	Upper	1950-06-30T23:59:59Z	-615513601
2004-07-03	End	Lower	2004-07-03T00:00:00Z	1088812800
2004-07-03	End	Upper	2004-07-03T23:59:59Z	1088899199

While "1950-06-XX/2004-07-03" is a good and compact way to represent a complex date structure there aren't any databases (yet) that know how to parse EDTF strings or make them searchable. By decomposing an EDTF string in to inner and outer ranges of dates and timestamps (specifically **Unix timestamps** `https://en.wikipedia.org/wiki/Unix_time`) we are able to easily store and query these values in any database that can index numeric values.

Additionally it is a model that allows people query for records using EDTF strings as their input parameters. For example if someone wanted to search for records during the "1950s" their query term would be the EDTF date string "195X". The outer limits of that date are `-631152000` and `-315619201` (1950-01-01T00:00:00Z and 1959-12-31T23:59:59Z, respectively). Assuming a database that has been indexed with numeric columns for "start" and "end" dates the

query for records in "the 1950s" becomes a simple greater-than and lesser-than operation like this, which would capture the record above whose date is encoded as "1950-06-XX/2004-07-03":

```
SELECT * FROM records WHERE start >= -631152000 AND end <= -315619201
```

Which is a lot easier than teaching your database what "the 1950s" are.

The EDTF specification does all the work of defining the rules and semantics for encoding complex and ambiguous dates in to well-defined and structured strings and the `go-edtf` package does the work of decomposing those strings in to values and flags that can be manipulated by computers.

The EDTF specification has a number of rules for expressing uncertainty, precision and other properties describing a date. These details are out of scope for this blog post but all of these properties are surfaced by the code in the `go-edtf` package. How those properties are stored or used in a given setting are decisions left to the organization consuming EDTF date strings.

1. ***https://github.com/sfomuseum/go-edtf*** `https://github.com/sfomuseum/go-edtf`

go-edtf-wasm

Token: Braniff International. Plastic, ink. Gift of the Captain John B. Russell Family, SFO Museum Collection. ***2012.147.135***

As the name suggests the `go-edtf` tools are written in the Go programming language. Not everyone wants to, or can, use Go for their programming needs so we have released a second tool to expose the core functionality of the `go-edtf` package as **WebAssembly** `https://webassembly.org/` binaries.

WebAssembly (wasm) is a standard that defines the rules for exporting code written in one programming language so that it can be safely consumed and executed by another programming language. Not all programming languages support building or running wasm binaries yet but one that does is Go. Another is JavaScript and is supported by all the major web browsers.

The second tool we've released is called **go-edtf-wasm** `https://github.com/sfomuseum/go-edtf-wasm` and it is code for building wasm binaries that expose specific pieces of

functionality from the `go-edtf` package. Today there is only one wasm binary that exports a `parse_edtf` function for parsing an EDTF string in to a JSON-encoded `EDTFDate` object, as described above. Here's an abbreviated example of how we might load the `parse.wasm` binary and then use it later on in a web application to parse an EDTF date entered in a form:

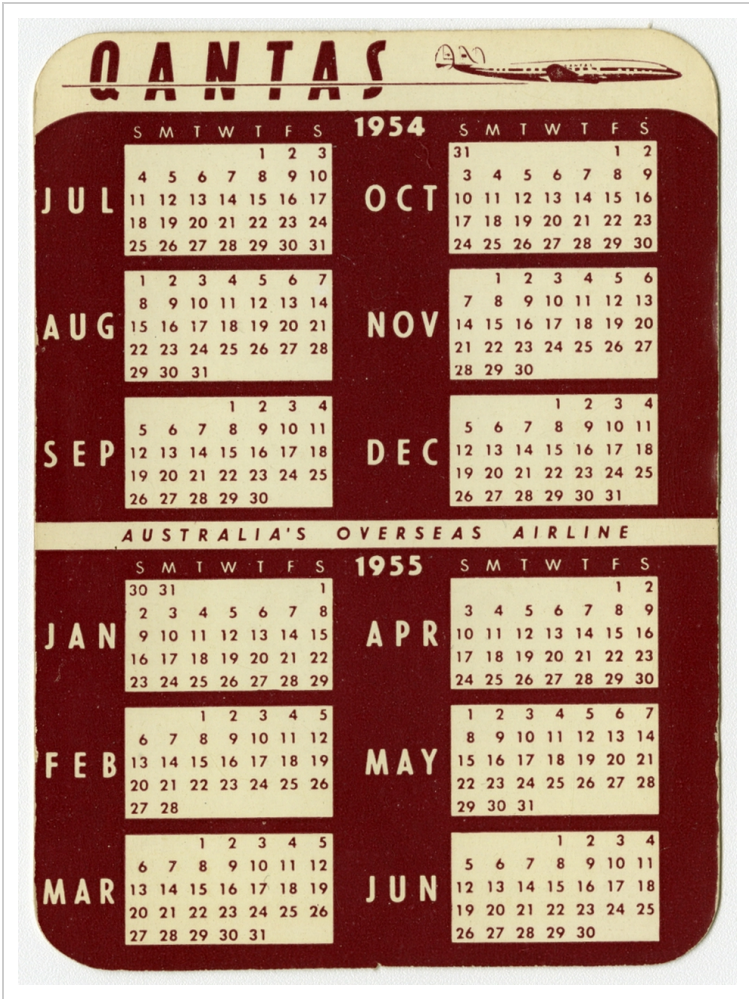
```
const go = new Go();

let mod, inst;

WebAssembly.instantiateStreaming(fetch("/wasm/parse.wasm"), go.importObject).then(
    async result => {
        mod = result.module;
        inst = result.instance;
        await go.run(inst);
    }
);

var edtf_el = document.getElementById("edtf");
var edtf_str = edtf_el.value;

var rsp = parse_edtf(edtf_str);
var edtf_date = JSON.parse(rsp)
```

And just like that all the code that we've written code in Go is available to tools not written in Go. A tool like `parse.wasm` can be useful for doing client-side validation of EDTF dates before submitting them to a search endpoint or for converting those dates into their numeric (or datetime) equivalents and passing those to servers that aren't able to run the native `go-edtf` code themselves.

We are just getting started investigating the different uses for WebAssembly binaries and are excited to see where they take us.

1. ***https://github.com/sfomuseum/go-edtf-wasm*** `https://github.com/sfomuseum/go-edtf-wasm`

go-edtf-http



Pocket calendar: Qantas Airways, 1954. Paper, ink. Gift of Thomas G. Dragges, SFO Museum Collection. ***2015.160.748***

The third tool we are releasing is **go-edtf-http** `https://github.com/sfomuseum/go-edtf-http` and provides convenience methods for creating **Go-language HTTP "handlers"** `https://golang.org/doc/articles/wiki/` for exposing functionality defined in `go-edtf` as machine-readable

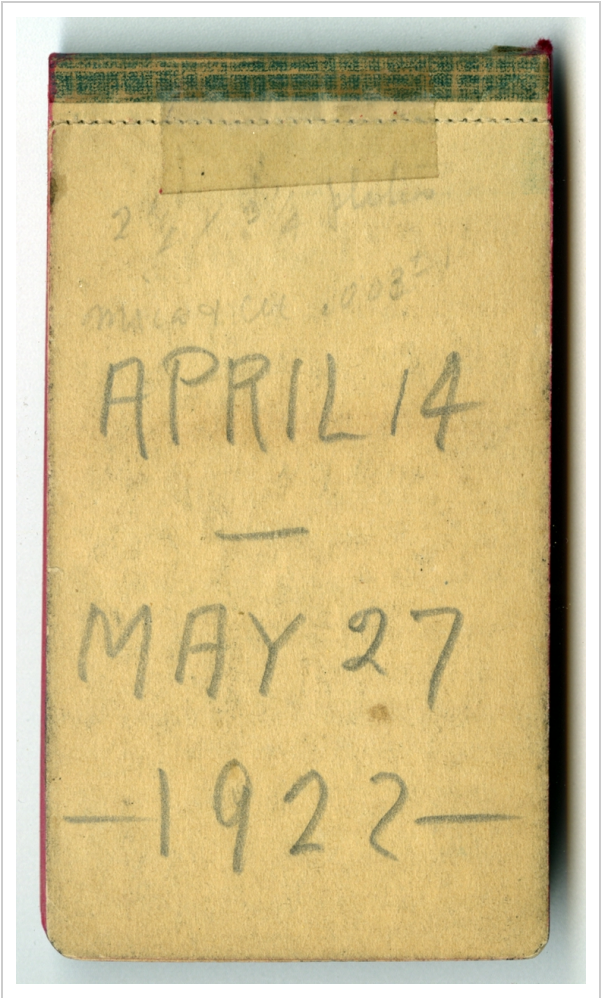
web services. For example, here is the simplest implementation of that, a simple API-over-HTTP web server:

```
import (
        "github.com/sfomuseum/go-edtf-http/api"
        "net/http"
)

func main() {

        api_parse_handler, _ := api.ParseHandler()
        api_valid_handler, _ := api.IsValidHandler()
        api_matches_handler, _ := api.MatchesHandler()

        mux := http.NewServeMux()
        mux.Handle("/api/parse", api_parse_handler)
        mux.Handle("/api/valid", api_valid_handler)
        mux.Handle("/api/matches", api_matches_handler)

        http.ListenAndServe("localhost:8080", mux)
}
```

```
$> curl -s 'http://localhost:8080/api/valid?edtf=Jan,%203%201985'
false

$> curl -s 'http://localhost:8080/api/valid?edtf=1985-01-03'
true

$> curl -s 'http://localhost:8080/api/matches?edtf=1985-01-03/1987'
{"level":0,"feature":"Time Interval"}
```

1. ***https://github.com/sfomuseum/go-edtf-http*** `https://github.com/sfomuseum/go-edtf-http`

go-sfomuseum-edtf



Logbook: Stanley Henry Page. Paper, ink. Gift of Charles Page, SFO Museum Collection. ***2010.174.023***

The fourth and final tool we've released is specific to SFO Museum. It is provided "as-is" in a spirit of sharing and generosity and as an example of how all the other tools can be combined in to an application tailor-fit for an organization (SFO Museum). Like the other tools `go-sfomuseum-edtf` is written in the Go

programming language and has two parts:

1. *Code and tools for converting strings written using SFO Museum's rules and conventions for encoding dates in to EDTF strings and* `EDTFDate` *objects.*

2. *A simple web application that allows people to enter SFO Museum date strings and convert them in to EDTF strings and* `EDTFDate` *objects. The application also exposes this functionality via machine-readable web services and uses the code in the* `go-edtf-http` *package for functionality that isn't specific to SFO Museum.*

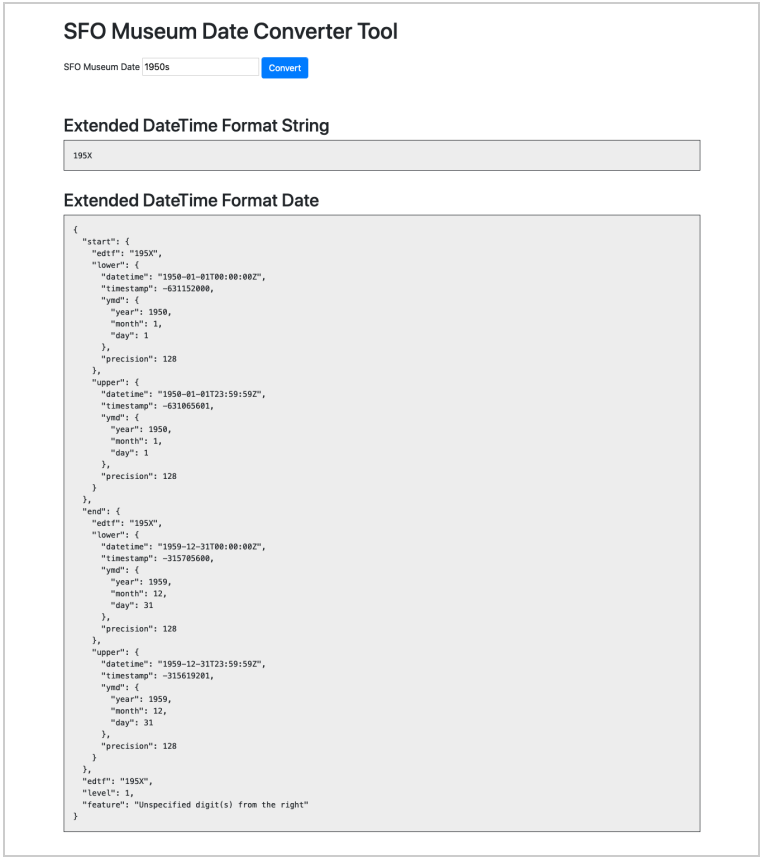
SFO Museum has an existing practice, a muscle-memory, around how dates are represented in our collection and it is unrealistic, impractical and impolite to suddenly ask everyone to adopt ETDF dates. Overall, EDTF might be a better representation for dates and if everyone adopted it we, as a community, could make "short work" of many inoperability problems involving time but it is unlikely that EDTF will be adopted immediately by most organizations.

Until then we need tools to meet the existing practice half-way and to be able to translate the dates already in use in to their EDTF equivalents. That is what `go-sfomuseum-edtf` aims to do for SFO Museum. Here's an example of some of **the patterns that SFO Museum uses** `https://github.com/sfomuseum/go-sfomuseum-edtf/blob/www/README.md#patterns` for dates and their EDTF representations:

SFO Museum date	**EDTF date**
04/1972	1972-04

c. 03/12/1984	1984-12-~03
early 1970s	1970-01/1970-04
c. early 1950s	1950-~01/1950-~04
Early 1960s	1960-01/1960-04
1930s	193X
c 1980s	~198X-01-01/~198X-12-31
c. 2018- 2020	~2018/~2020

And here's a screenshot of the web application we built to convert SFO Museum dates in to EDTF dates:



The tool itself is available on the **Mills Field website** `https://millsfield.sfomuseum.org/edtf` at: **https://millsfield.sfomuseum.org/edtf/** `https://millsfield.sfomuseum.org/edtf/`

This tool exists to demonstrate 1) How we can use EDTF dates without asking staff to enter EDTF dates or change their practice and 2) To show staff how to translate (a) to (b) when it's necessary. I also anticipate it will be useful for testing new date strings we adopt or when we discover patterns that aren't handled by the code yet.

1. ***https://github.com/sfomuseum/go-sfomuseum-edtf*** `https://github.com/sfomuseum/go-sfomuseum-edtf`

Logbook: Stanley Henry Page. Paper, ink. Gift of Charles Page, SFO Museum Collection. ***2010.174.018***

The Extended DateTime Format may not be the "single biggest" improvement, or change, to the cultural heritage sector in recent memory but I believe that once more people realize what it is and what it does, and the tools that exist for working with EDTF dates, that it will be a big deal. It is a big deal because these tools represent

a bridge between the necessary ambiguity of representing time and the practical mechanics of searching for things by time in a... well, timely manner.

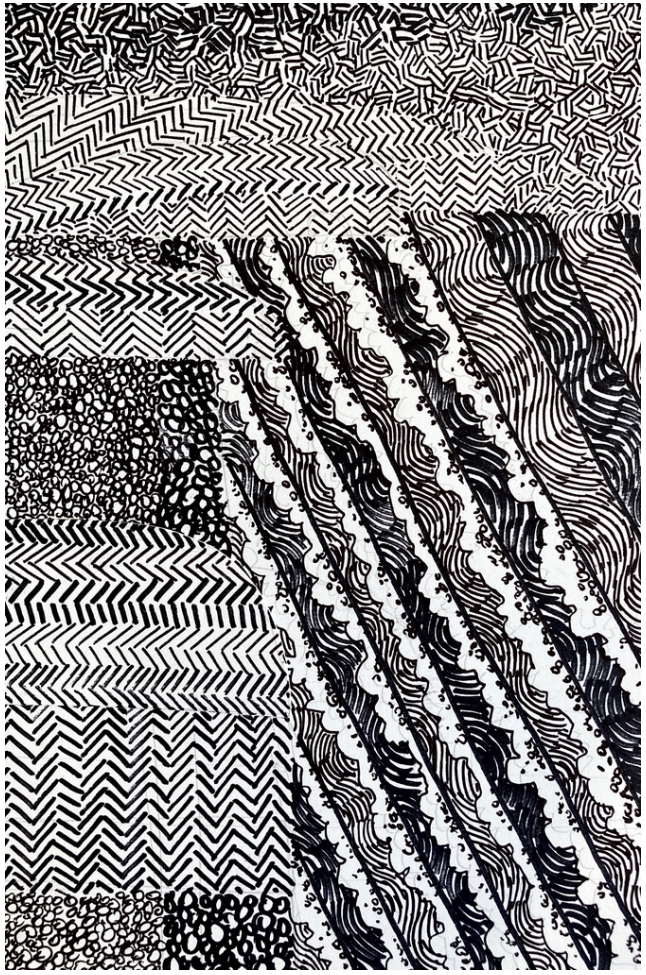
The **sfomuseum/go-edtf packages** `https://github.com/sfomuseum?q=edtf` are not the first or only **software implementations of EDTF** `https://www.loc.gov/standards/datetime/implementations.html` but we hope they will be a valuable contribution and another step towards building a common toolbox of **small focused tools** `https://millsfield.sfomuseum.org/blog/2020/08/04/small-tools/` by and for the cultural heritage community.

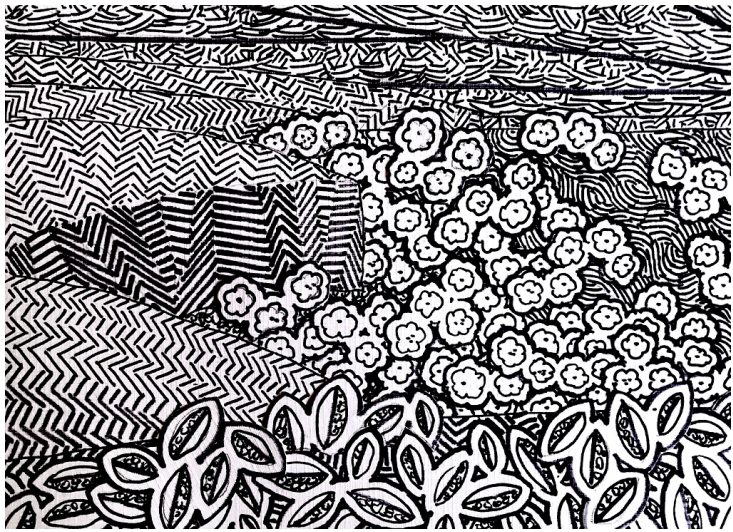
2021-01-14

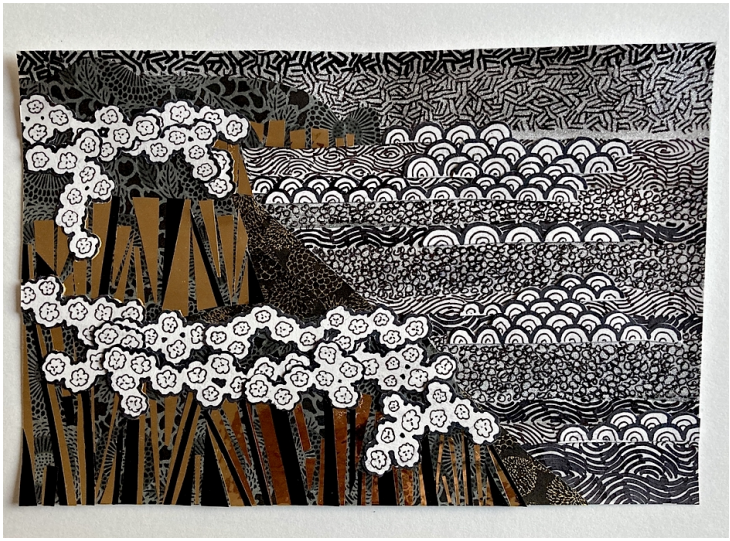
days of fog and sparkly paint

Three views from Mussel Rock

Three views from Mussel Rock







Mussel Rock is a short ways up the coast from **Mori Point** https://www.aaronland.info/weblog/2020/09/22/allusion/#mori-point .

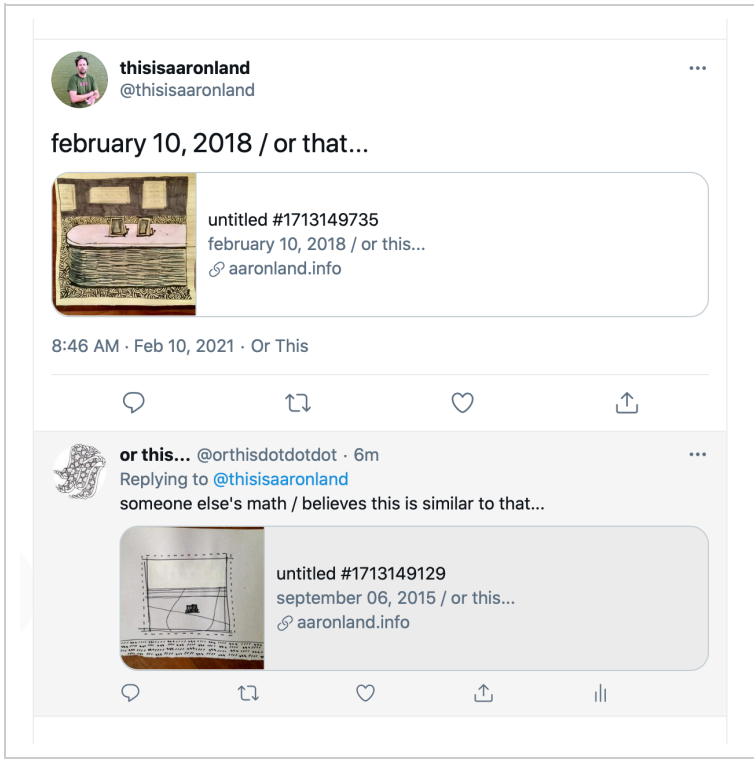
2021-01-19

the sky we live under

Someone else's math

Six views from Ocean Beach

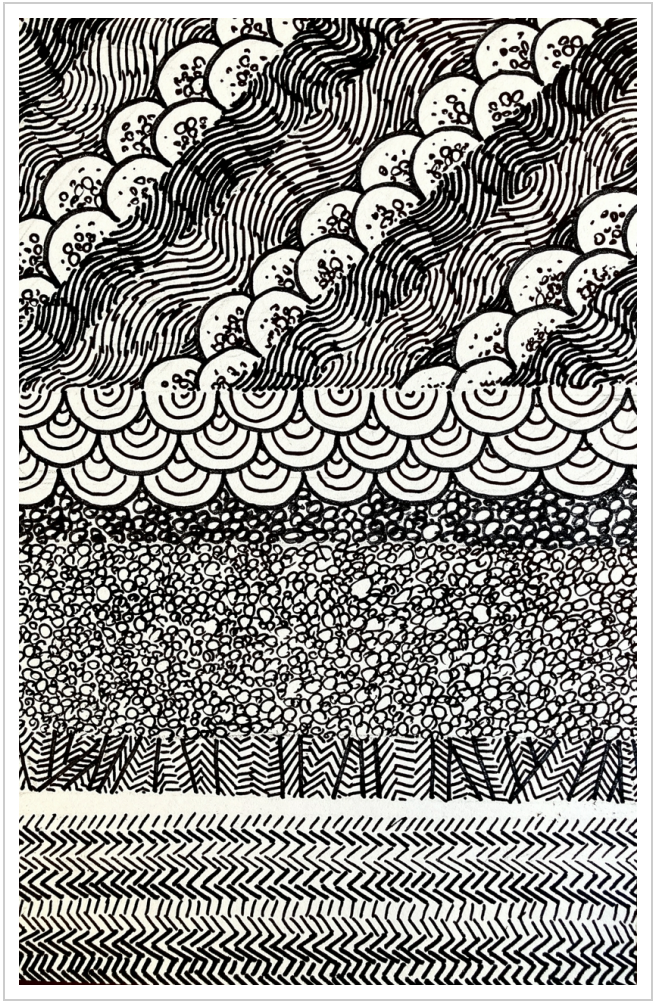
Someone else's math



Posting here for the sake of posterity because this is the story I want to remember. Also that day **in 2018** `https://aaronland.info/orthis/171/314/973/5/` when I went for what I thought was going to be a friendly, casual conversation and was ambushed with whiteboard exercises.

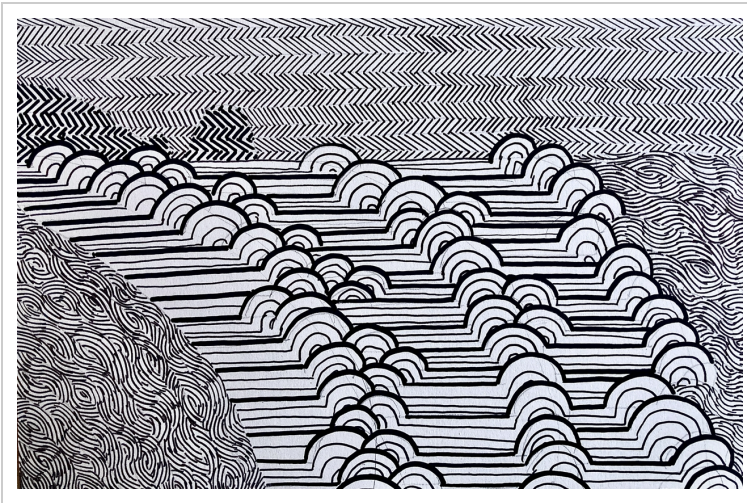
2021-02-10

Six views from Ocean Beach

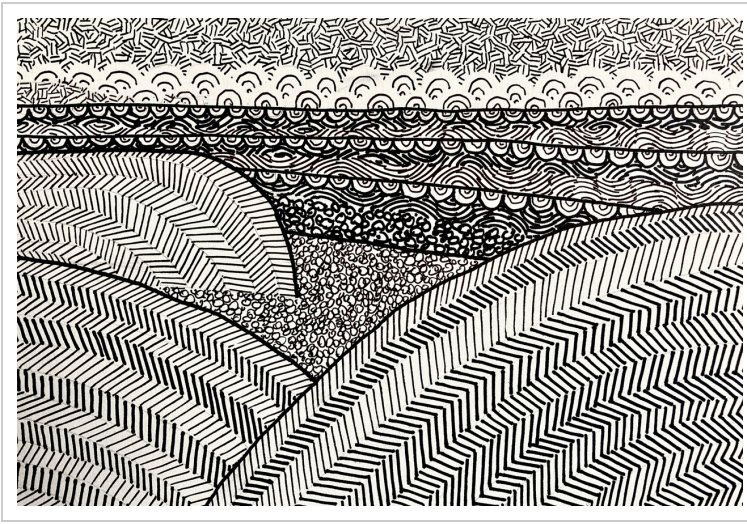


2020

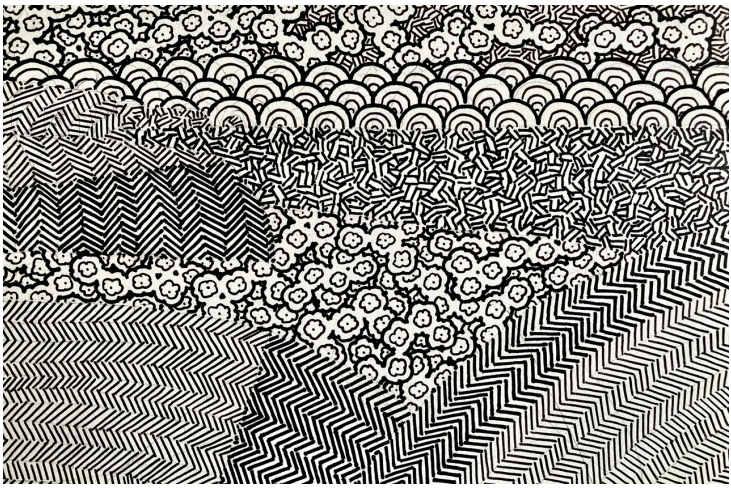
2020



2021



2021



2021

2021

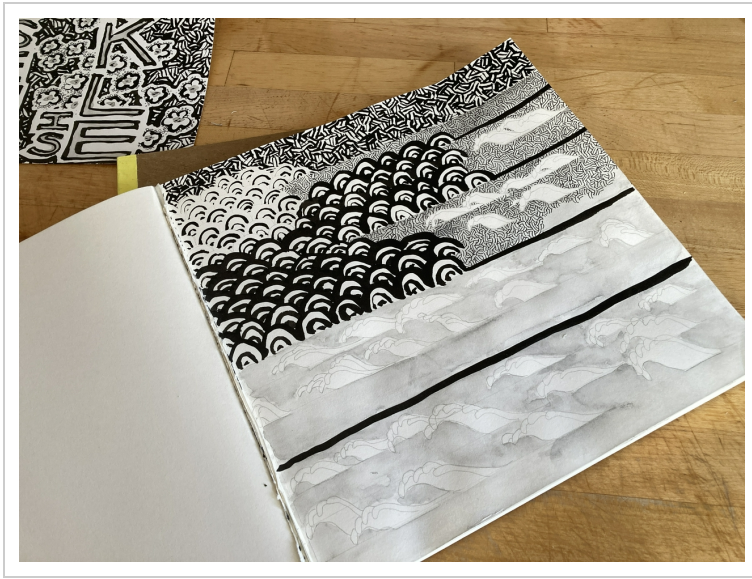
Ocean Beach is a short ways up the coast from **Mussel Rock** https://www.aaronland.info/weblog/2021/01/19/days/#views . Prints of the first drawing **are available for sale** https://orthis.myshopify.com/collections/frontpage/products/ocean-beach-2020 . Prints of the others, as well as the original drawings themselves, may go **on offer** https://aaronland.info/orthis/forsale/ shortly.

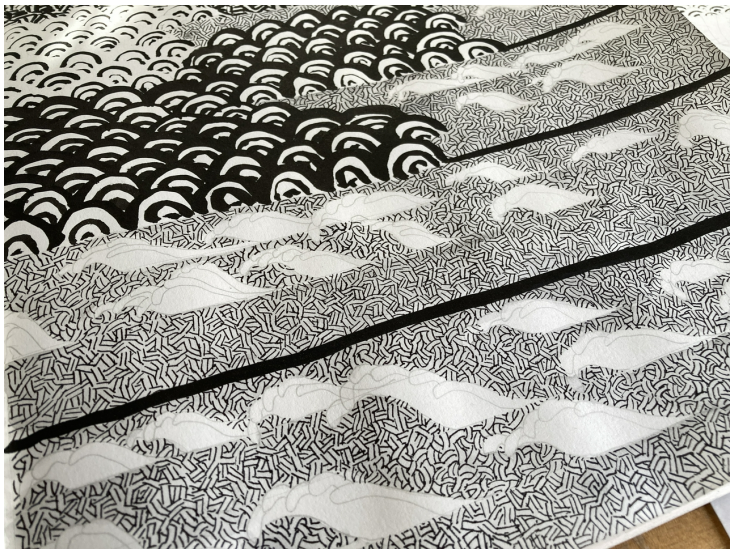
2021-02-08

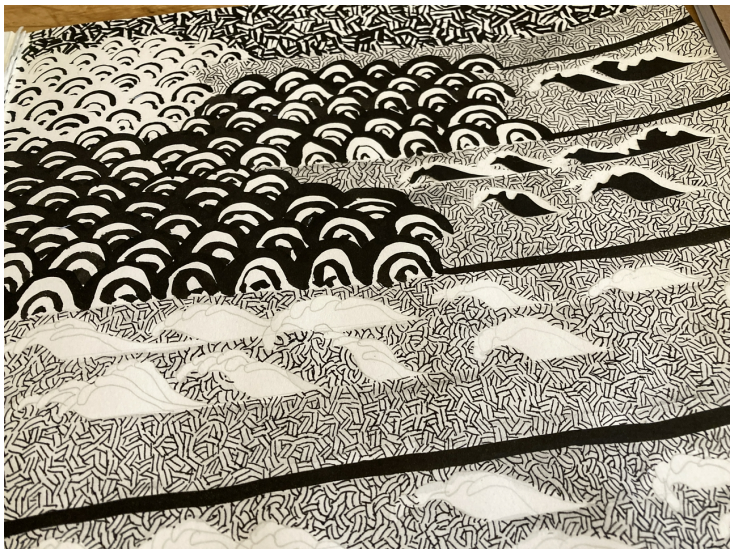
generative scarcity

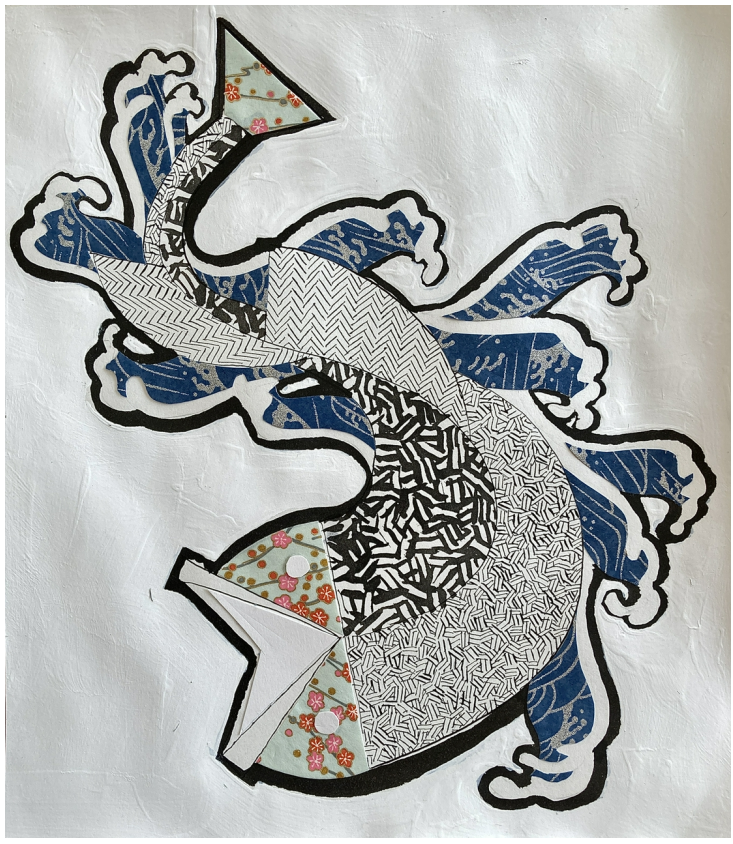
I still like cut outs

I still like cut outs









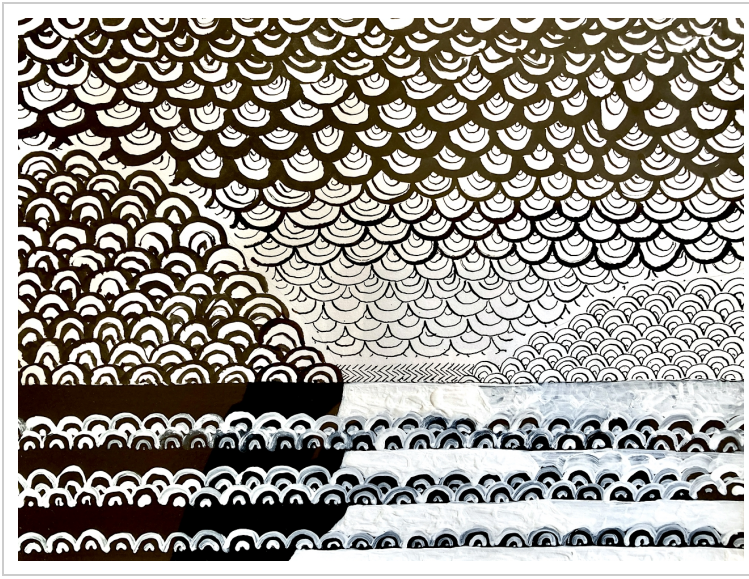
See also: **I made some drawings. Then I cut them in to pieces.** `https://aaronland.info/weblog/2020/02/24/scrolling/#cut-outs`

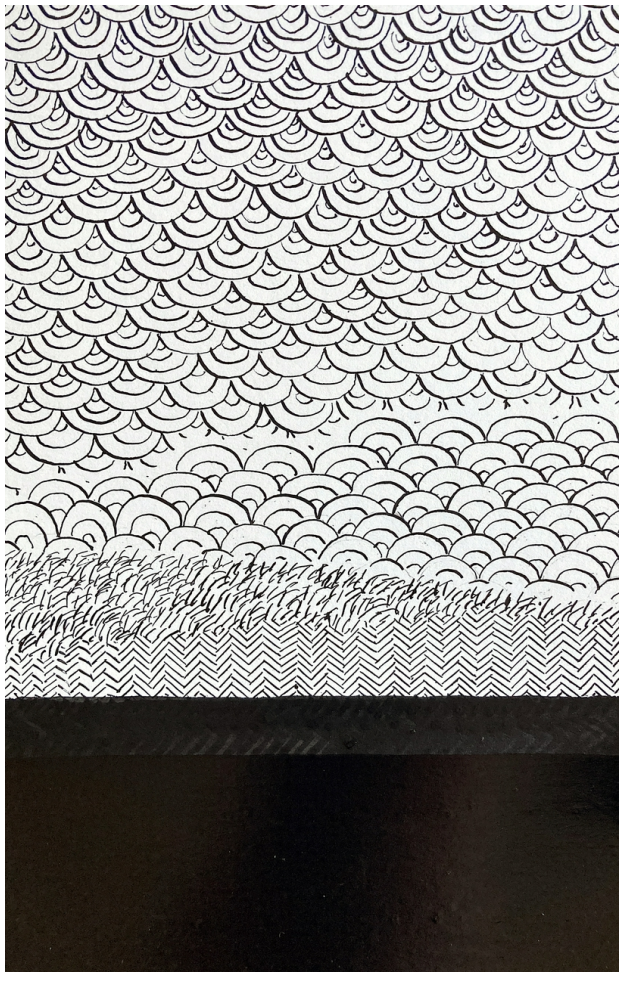
2021-02-24

slow computing

Two views from Estero Americano (2021)

Two views from Estero Americano (2021)





Estero Americano is up the coast from **Ocean Beach** https://www.aaronland.info/weblog/2021/02/08/sky/#views and down the coast from **Timber Cove** https://www.aaronland.info/weblog/2019/12/29/kitchen/#timbercove .

2021-03-04

escape velocity as asset class

cheap money, so to speak

cheap money, so to speak

I don't have much to say on the subject of non-fungible tokens (NFTs). They are all the rage these days mostly, it seems, because they are understood as a fast and furious speculative window for achieving a kind of financial escape velocity. Compounding the issue is the idea that "escape velocity" has become, broadly speaking, the only way we know how to measure "success" anymore. Recently NFTs intersected with the still nascent push to make cultural heritage collections, and in particular imagery of those collections, open and accessible under the most liberal and permissive terms possible. You won't believe, as they say, what happened next.

There's been quite a lot of anger and concern about people creating and selling NFTs derived from open-access cultural heritage collections, notably the

Rijksmuseum `https://www.rijksmuseum.nl` . The subject came up recently in a forum for museum people. I am posting an annotated version of what I said there mostly to have a record of it for myself and because I can imagine having to reference it in the future.

> I am struggling to see how they (NFTs) in a museum context are any different from postcards in the shop. Given the sector's terrible history with image rights and the dream of having the proverbial "Mona Lisa postcard" that will be so popular as to fund the rest of the museum I am... skeptical.

If you don't live and breathe the business of museums that last comment, about the "Mona Lisa postcard", might seem a little obscure. One of the principal roadblocks to cultural heritage institutions publishing openly licensed versions of their collection imagery is the fear of a loss of revenue from the sale of that imagery. The reality is that the sales of image rights in museums are so paltry that the costs to administer the restrictions around them exceed any profits they generate. There are, it's true, a small handful of museums with one or two works whose narrative pedigree ensures consistent sales and something like a profit. It doesn't matter that these handful of images are the outliers. Everyone *wants to believe* that their collections are just as popular.

> The Rijksmuseum famously said they hoped people had toilet paper printed with their collection on it.

They really did. I no longer remember which conference it was at but I was in the audience when it happened.

> Since the Rijksmuseum collection was **released under a public domain license** `https://www.rijksmuseum.nl/en/research/conduct-research/data/policy` said NFT-company isn't doing anything wrong and is actually under no obligation to share revenue(s) with the museum.
>
> I said earlier that NFT + museums smelled like postcards of old but there is a plausible argument that it's just a fancy tip jar, as in people buy some part of an artwork but the only "thing" they are actually "buying" is a warm and fuzzy feeling. A warm and fuzzy feeling that, technically, you could resell in the same way that we got ourselves in to the subprime mortgage mess...

A bit later I summarized all of that as:

> Public domain means you don't get to choose, the Rijksmuseum went in to this with eyes open, museums have **a terrible track record when it comes to image rights** `https://aaronland.info/weblog/2017/02/20/mostly/#museumnext` so who are we to judge and "basically a glorified tip-jar".

And because I couldn't help myself, I went on to say:

> This dynamic has been playing out in the software world since the advent of "free" or "open source" software, most recently with the efforts of companies like **Elastic(search)** https://www.elastic.co/blog/why-license-change-AWS that provide hosted services for the free software they maintain to prevent other companies (notably **Amazon** https://aws.amazon.com/blogs/opensource/stepping-up-for-a-truly-open-source-elasticsearch/) from hosting their own cheaper versions.
>
> You also see it in the never-ending debate around the licensing of OpenStreetMap data which switched from being distributed under a Creative Commons license to the **Open Database license** https://en.wikipedia.org/wiki/Open_Database_License which is effectively a viral "copyleft" license aimed to prevent commercial enterprises from reaping the rewards of the (OSM) community's work without giving anything back.
>
> Ever since the introduction of the **GNU Public License** https://www.gnu.org/licenses/gpl.html for software, which judging by the tenor of the comments about NFTs is what people are looking for for cultural heritage works, there have been the **MIT** https://en.wikipedia.org/wiki/MIT_License and **BSD** https://en.wikipedia.org/wiki/BSD_licenses software licenses that say: Do whatever you want with this code, including using it in commercial

> projects or for otherwise questionable uses.
>
> But fundamentally, putting things in the public domain means a) You don't get to choose and b) You believe that the benefit of making these things available outweighs their occasional misuse. Which has essentially been the message of both the Rijksmuseum and **the Smithsonian** `https://creativecommons.org/2020/02/27/smithsonian-releases-2-8-million-images-data-into-the-public-domain-using-cc0/` (not to mention **Europeana** `https://data.europa.eu/euodp/en/about`) in releasing their data.

It's worth repeating: You believe that the benefit of making these things available outweighs their occasional misuse.

> Even though it doesn't really pass the smell-test the people issuing NFTs around this stuff aren't doing anything "wrong". The whole thing would be a good opportunity for the sector to point out the folly of the (NFT) endeavour but I wonder whether the sector won't instead get swept away by the jealously and envy of cheap money, so to speak.

On that last point there is also a whole other conversation to be had about whether or not there is any meaningful distinction between NFTs for works in our collections and **named or sponsored galleries** `https://www.nytimes.com/2019/07/18/arts/sackler-`

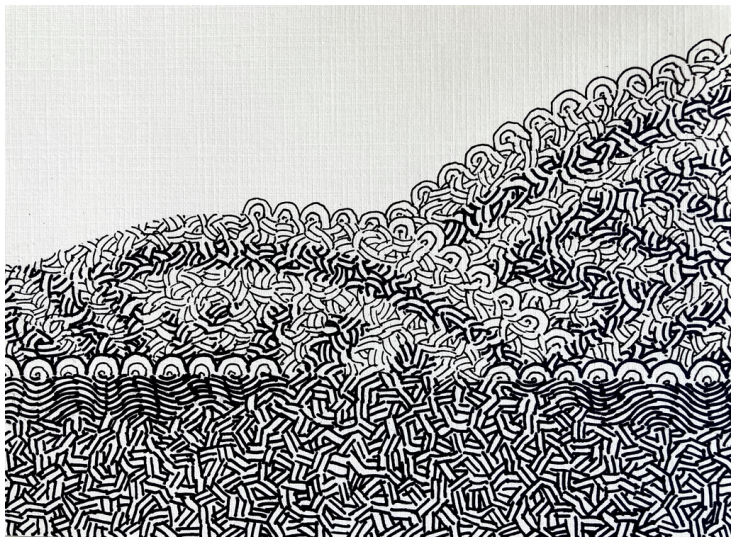
`family-museums.html` . That is probably still too raw for most museums to want to talk about in public, though.

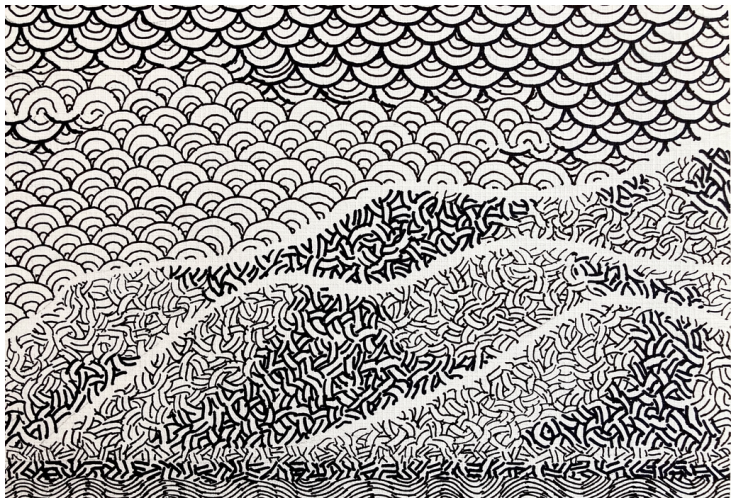
2021-03-19

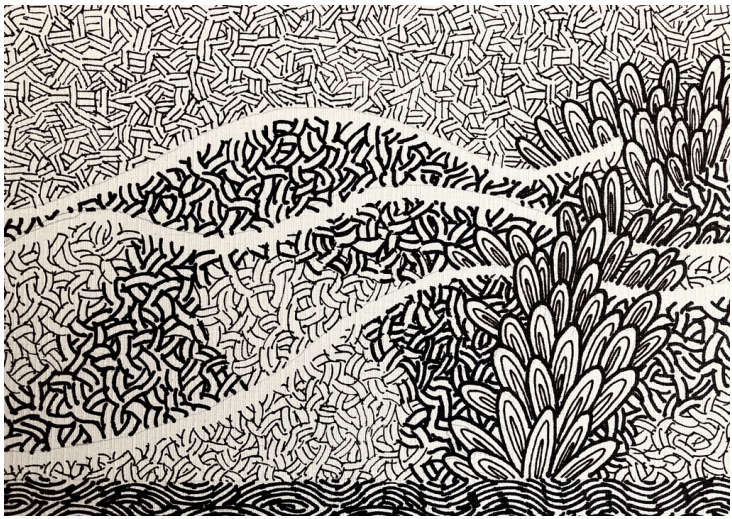
one year on

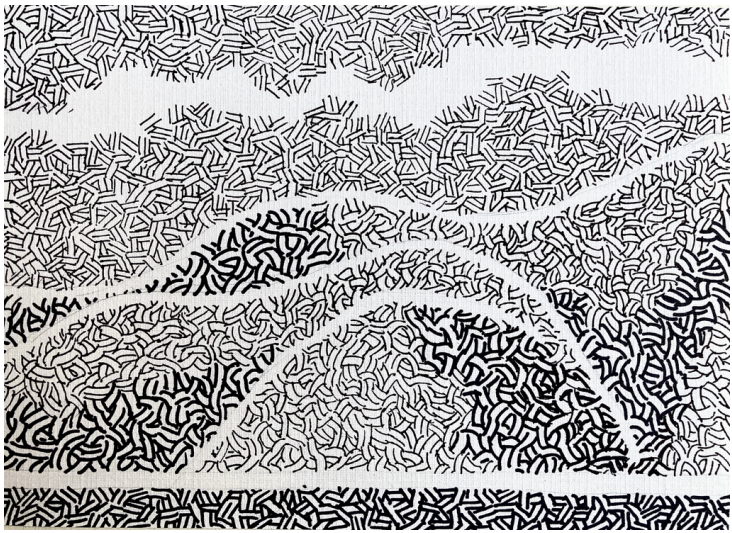
Six views from the Russian River (2021)

Six views from the Russian River (2021)









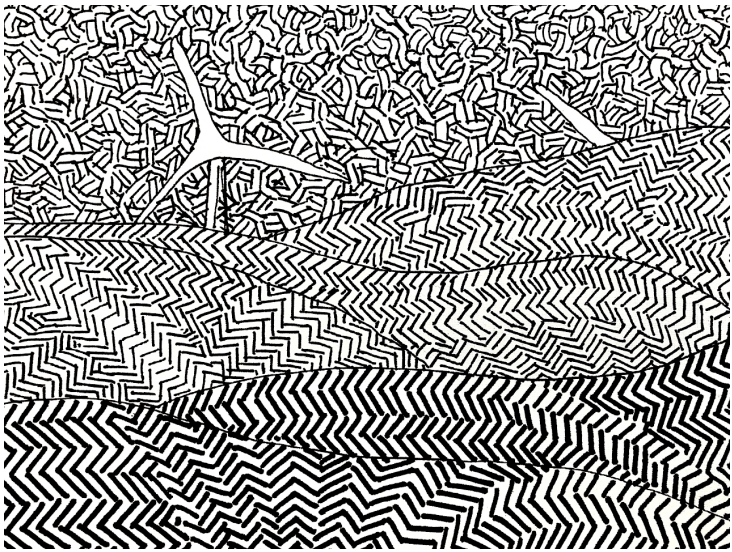
The Russian River is up the coast from **Estero Americano** https://www.aaronland.info/weblog/2021/03/04/slow/#views and down the coast from **Timber Cove** https://www.aaronland.info/weblog/2019/12/29/kitchen/#timbercove .

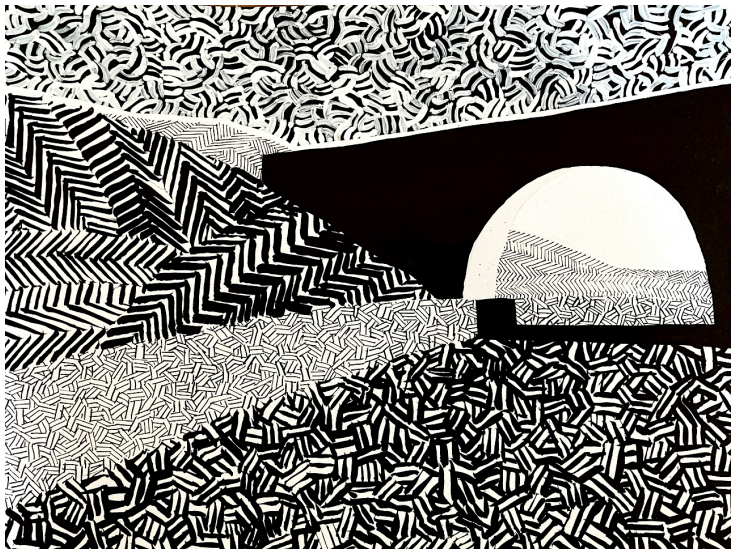
2021-03-22

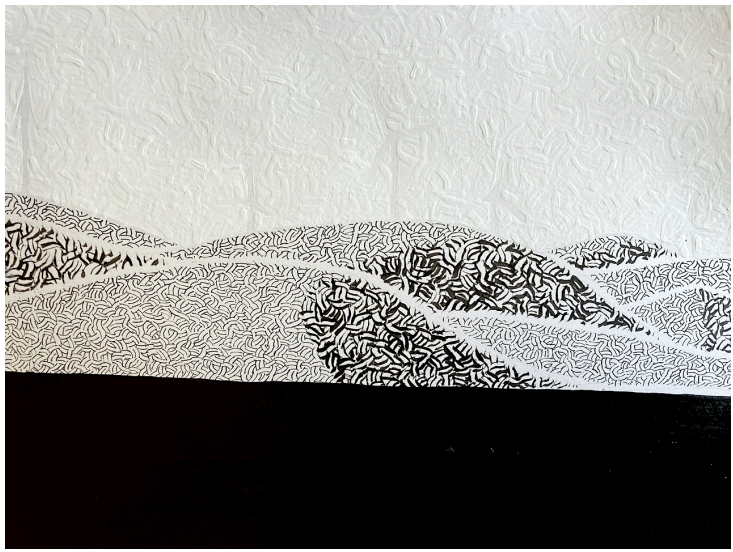
I guess waves are like the ocean's thumbs, math

Three views from the 580 East (2021)

Three views from the 580 East (2021)







These were around Livermore, where the windmills give way to a short stretch of low hills and cattle-grazing that level out in to mile after mile of distribution centers.

2021-04-09

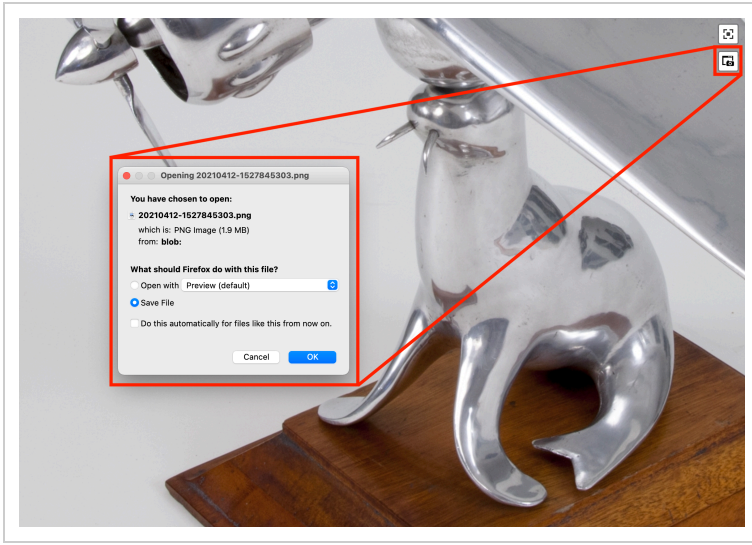
things I have written about elsewhere #20210414

Updating EXIF metadata in JavaScript (and...

Updating EXIF metadata in JavaScript (and WebAssembly)

This was **originally published on the Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2021/04/14/exif/` , in April, 2021.

Over the last year SFO Museum has been hard at work on the tools it uses to build and display "**zoomable images** `https://millsfield.sfomuseum.org/blog/tags/zoomable` ". These tools allow a static, fixed-size image to be converted in to a tiled and "zoomable" version that can be used to pan around and highlight specific details. One of the handy features with zoomable images is the ability to save the visible area of a "zoomed" image to a new file, by clicking the camera icon in the top right-hand corner of the viewport.



*Screenshot of the zoomable image camera button creating a new image with a crop of my favourite collection object "**Tabletop airplane with walrus**".*

One question that's been raised about the camera/save button in zoomable images is whether or not the new image contains, or preserves, existing EXIF metadata information stored in the original file. The answer yesterday was: No. The answer today is: Not yet, but only because we haven't enabled it and we will do that soon.

Today we are releasing the **go-exif-update** `#` package. It is a thin wrapper around Dustin Oprea's **go-exif** `https://github.com/dsoprea/go-exif` and **go-jpeg-image-structure** `https://github.com/dsoprea/go-jpeg-image-structure` packages designed to make assigning or updating EXIF properties in JPEG files a little bit easier. Importantly our package includes tools for compiling the code in to a **WebAssembly** `https://developer.mozilla.org/en-US/docs/WebAssembly` (wasm) binary for updating EXIF data in JavaScript applications.

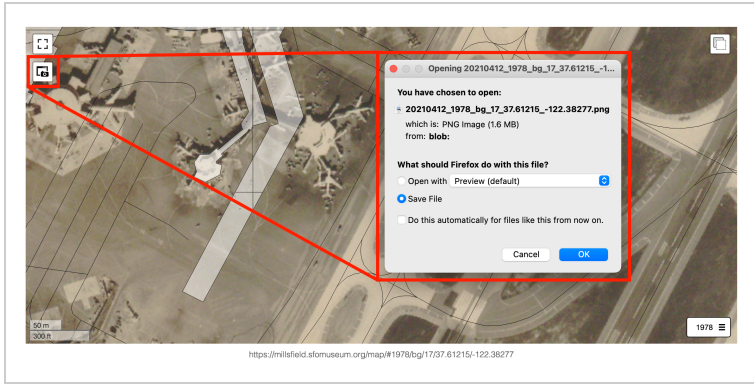
In the **Tools for Complex and Ambiguous Dates at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/01/14/edtf/` blog post I wrote that:

> WebAssembly (wasm) is a standard that defines the rules for exporting code written in one programming language so that it can be safely consumed and executed by another programming language. Not all programming languages support building or running wasm binaries yet but one that does is Go. Another is JavaScript and is supported by all the major web browsers.

This is the second tool we've built using wasm (the first was a tool to **parse ExtendedDateTime Format date strings** `https://github.com/sfomuseum/go-edtf-wasm`) and it is an approach to modular and portable functionality that continues to be promising. The ability to update EXIF data in images in JavaScript means that we can add the relevant metadata, like a credit line or a permanent URL, to zoomable images as they are created and before they are downloaded. Here's an abbreviated version of what that looks like in (JavaScript) code:

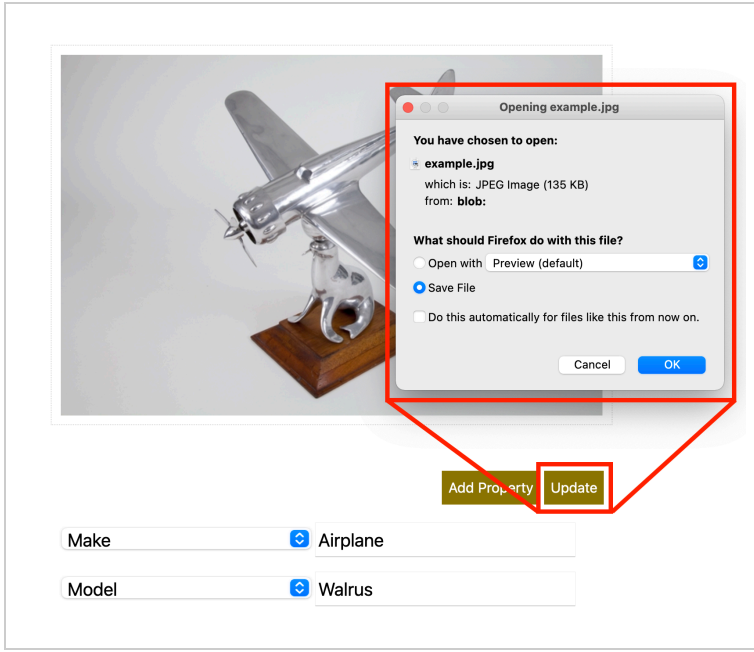
```
var update = { "Artist": "Bob" };
var enc_update = JSON.stringify(update);

var img = document.getElementById("image");

var canvas = document.createElement("canvas");
canvas.width = img.width;
canvas.height = img.height;
var ctx = canvas.getContext("2d");
ctx.drawImage(img, 0, 0);
var b64_img = canvas.toDataURL("image/jpeg", 1.0);

// This is where the "magic" happens
var rsp = update_exif(b64_img, enc_update);

var blob = dataURLToBlob(rsp);
saveAs(blob, "example.jpg");
```

As of this writing the majority of **EXIF tags** `https://exiftool.org/TagNames/EXIF.html` are not yet supported. Currently **only EXIF tags that are stored as strings** `https://github.com/sfomuseum/go-exif-update#supported-tags` are supported. I am still trying to get familiar with the requirements of the `go-exif` package so this is just a first step. I am hoping to enable the writing of numeric EXIF tags soon because the other place where SFO Museum has a save-to-file camera button that would benefit from dynamically-written EXIF tags is the **historical maps** `https://millsfield.sfomuseum.org/map/#1978/bg/17/37.61215/-122.38277` webpage.



For example, **this snapshot of a map of SFO in 1978** `https://millsfield.sfomuseum.org/map/#1978/bg/17/37.61215/-122.38277` should have both a URL and GPS information embedded in its EXIF metadata. That's possible to do now!

Demonstration



To demonstrate adding dynamically-generated EXIF metadata to images we've included the `update_exif.wasm` binary with this blog post. We've actually included *two* WebAssmebly binaries, one to update the EXIF data and another to return a list of EXIF tags supported by the `go-exif-update` package. We used this second binary to pre-populate a select menu with available tag names.

Once the two wasm binaries have finished loading you should see an `Add property` button beneath the image of the **tabletop airplane with walrus** `https://millsfield.sfomuseum.org/objects/1511908311/` . Each time you click the button a new row, containing a menu of EXIF tag names and an input field for a value to assign to that tag, will be created.

Loading

To finish writing the new EXIF tags to the image simply click the `Update` button. Once complete your web browser will prompt you to save the new image to your computer and you can inspect the file's EXIF data with the tool of your choice. For example using the **exiv2** `https://www.exiv2.org/` command-line tool:

```
$> exiv2 -pa ~/Downloads/example.jpg
Exif.Image.Make                              Ascii       9  Airplane
Exif.Image.Model                             Ascii       7  Walrus
```

This is only a first release of the code to update EXIF metadata in JavaScript and it is expected that there will be a few "gotchas" before it's considered stable for general use. One important consideration is that the `update_exif.wasm` binary is sufficiently large (7.2MB) that it can have a noticeable impact on the time it takes to load a webpage. This is especially true if WebAssembly binaries aren't cached by web browsers. If you

have any experience with these issues we'd **love to hear about your experiences** `https://github.com/sfomuseum/go-exif-update/issues` .

1. ***https://github.com/sfomuseum/go-exif-update*** `https://github.com/sfomuseum/go-exif-update`

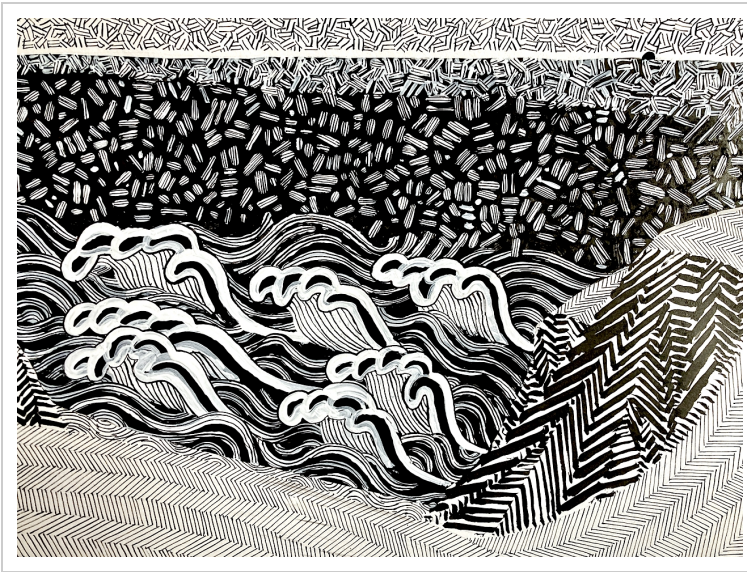
In the meantime we're going to start thinking about how best to integrate this new functionality with our **zoomable images code** `https://github.com/sfomuseum/js-zoomable-images` and we'll keep you posted on our progress.

2021-04-14

interpretation pirates

Three views from Mussel Rock (April, 2021)

Three views from Mussel Rock (April, 2021)



I still don't know whether I am happy with any of these drawings, so **I guess we'll see where things stand in a year's time** `https://aaronland.info/weblog/2020/06/08/covert/#date` .

2021-04-16

things I have written about elsewhere #20210426

Updating (and reverse-geocoding) GPS EXIF metadata

Updating (and reverse-geocoding) GPS EXIF metadata

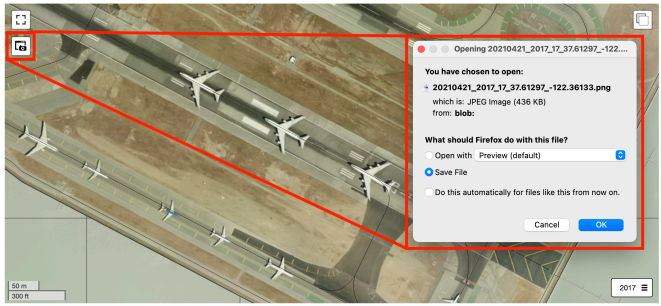
This was **originally published on the Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2021/04/26/exif-gps/` , in April, 2021.

#OnThisDay in 1968, the first 747 rolled out of Boeing's Everett plant. Have you ever been to the Boeing assembly building in Everett, Washington? #avgeek ***Twitter, September 30, 2018***

In the last blog post **Updating EXIF metadata in JavaScript (and WebAssembly)** `https://millsfield.sfomuseum.org/blog/2021/04/14/exif/` I wrote:

> Currently only EXIF tags that are stored as strings are supported [when dynamically updating EXIF information in JavaScript]. I am still trying to get familiar with the requirements of the go-exif package so this is just a first step. I am hoping to enable the writing of numeric EXIF tags soon because the other place where SFO Museum has a save-to-file camera button that would benefit from dynamically-written EXIF tags is the historical maps webpage.

I am happy to say this is now enabled on the **Mills Field historical maps webpage** `https://millsfield.sfomuseum.org/map` . For example when I save an image of this view of **the runways at SFO** `https://millsfield.sfomuseum.org/map/#2017/17/37.61297/-122.36133` , from 2017:



The newly created image's EXIF data contains not only the date and the permanent URL for the map but also the latitude and longitude coordinates for the map's center point:

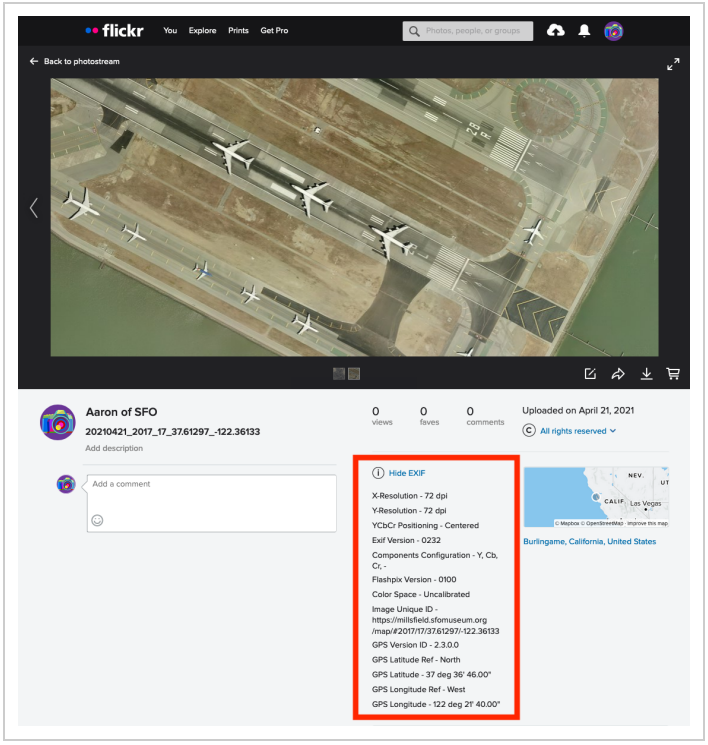
```
$> exiv2 -pa ~/Downloads/20210421_2017_17_37.61297_-122.36133.png
```

```
Exif.Image.GPSTag                           Long        1  76
Exif.GPSInfo.GPSLatitude                    Rational    3  37deg 36' 46"
Exif.GPSInfo.GPSLatitudeRef                 Ascii       2  North
Exif.GPSInfo.GPSLongitude                   Rational    3  122deg 21' 40"
Exif.GPSInfo.GPSLongitudeRef                Ascii       2  West
Exif.Image.DateTime                         Ascii       9  20210421
Exif.Image.ExifTag                          Long        1  178
Exif.Photo.ImageUniqueID                    Ascii      66  https://millsfield.sfomuseum.
Exif.Image.DateTimeOriginal                 Ascii       5  2017
```

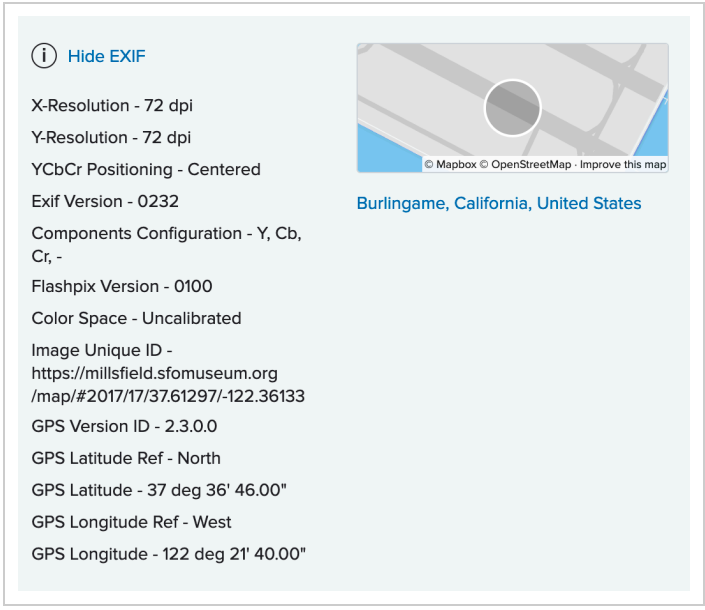
This new functionality to write GPS tags and other EXIF properties encoded as "rational" values is available in the `v0.0.3` (or higher) release of the **go-exif-update** `https://github.com/sfomuseum/go-exif-update` package:

1. ***https://github.com/sfomuseum/go-exif-update*** `https://github.com/sfomuseum/go-exif-update`

In order to test that these new GPS tags were being encoded correctly I **uploaded the map image to Flickr** `https://www.flickr.com/photos/161215698@N03/51131288670` which will **automatically read location information from EXIF tags** `https://www.aaronland.info/weblog/2015/02/24/effort/#holodeck` and, when present, place your photo on a map.



This screenshot of **our map image on Flickr** `https://www.flickr.com/photos/161215698@N03/51131288670` shows that the EXIF data was successfully imported and the image is geotagged at the end of **runway 10R/28L** `https://millsfield.sfomuseum.org/id/1730008749/` . Curiously, Flickr thinks the end of runway `10R/28L` is in **Burlingame** `https://spelunker.whosonfirst.org/id/85922537` .



Looking at their map I was reminded that in the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` blog post I wrote:

> This blog post is meant to be a quick show-and-tell to demonstrate two of the tools used to make all of this possible, one for creating spatial databases from arbitrary Who's On First records and the other for performing spatial queries using that data.

With that in mind, I made a set of tools, called **go-flickr-pip** `https://github.com/sfomuseum/go-flickr-pip` , that call the **Flickr API** `https://flickr.com/services/api` to retrieve geotagged

photos and then calls the local reverse-geocoding tools (described in the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` blog post and running on my laptop) to resolve where each one of those photos is located at SFO.

Negative: San Francisco International Airport (SFO), South Terminal. Negative. Transfer, SFO Museum Collection. ***2011.032.2638***

The first step was to create a new SQLite database from the **sfomuseum-data-architecture** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` data using the **wof-sqlite-index-features** `https://github.com/whosonfirst/go-whosonfirst-sqlite-features-index` tool:

```
$> ./bin/wof-sqlite-index-features \
        -all \
        -dsn /usr/local/data/sfomuseum-architecture.db \
        /usr/local/data/sfomuseum-data-architecture/

2021/04/22 12:49:41 time to index paths (1) 2.702893585s
```

Then I used that database as the data source for a local instance of the **go-whosonfirst-spatial-www-sqlite** `https://github.com/whosonfirst/go-whosonfirst-spatial-`

`www-sqlite` server running on my laptop:

```
$> ./bin/server \
       -spatial-database-uri 'sqlite://?dsn=/usr/local/datasfomuseum-architecture.db'

2021/04/22 12:50:29 time to index paths (0) 25.819µs
12:50:29.601179 [server] STATUS finished indexing in 425.886µs
```

The server exposes a simple web API for "reverse-geocoding" a point to one or more places which contain that point. For example, here's what the server returns for the coordinates assigned to our image of the map from 2017:

```
$> curl -s -XPOST http://localhost:8080/api/point-in-polygon -d '{"latitude": 37.612777,
{
  "places": [
    {
      "wof:id": "1730008749",
      "wof:parent_id": "102527513",
      "wof:name": "RUNWAY 10R/28L",
      "wof:country": "US",
      "wof:placetype": "custom",
      "mz:latitude": 37.61900555744979,
      "mz:longitude": -122.37573763583393,
      "mz:min_latitude": 37.6114694151077,
      "mz:min_longitude": 37.61900555744979,
      "mz:max_latitude": -122.39327941811621,
      "mz:max_longitude": -122.35819198079813,
      "mz:is_current": 1,
      "mz:is_deprecated": 0,
      "mz:is_ceased": 0,
      "mz:is_superseded": 0,
      "mz:is_superseding": 0,
      "edtf:inception": "",
      "edtf:cessation": "..",
      "wof:supersedes": [],
      "wof:superseded_by": [],
      "wof:belongsto": [
        102527513,
        102191575,
        85633793,
        102087579,
        85922583,
        554784711,
        85688637,
        102085387
      ],
      "wof:path": "173/000/874/9/1730008749.geojson",
      "wof:repo": "sfomuseum-data-architecture",
      "wof:lastmodified": 1619054586
    }
  ]
}
```

SFO Museum's own **architecture data** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` tells us that this point is contained by runway 10R/28L rather than Burlingame.

Finally I ran the `pip` tool in the **go-flickr-pip** `https://github.com/sfomuseum/go-flickr-pip` package to reverse-geocode geotagged photos in Flickr using SFO Museum data. This is what the output looks like for publicly-viewable photos in the **aaronofsfo** `https://www.flickr.com/photos/161215698@N03/` Flickr account:

```
$> go run -mod vendor cmd/pip/main.go \
        -client-uri 'oauth1://?consumer_key=...&consumer_secret=...' \
        -param method=flickr.photos.search \
        -param user_id=161215698@N03 \
        -param has_geo=1 \
        -param extras=geo

photo_id,latitude,longitude,whosonfirst_id,whosonfirst_name,whosonfirst_placetype
51130478394,37.615555,-122.388889,1729792681,International Terminal Connector,concourse
51130478394,37.615555,-122.388889,1729792579,International Terminal,wing
51130478394,37.615555,-122.388889,1729792387,SFO Terminal Complex,building
51130478394,37.615555,-122.388889,1729792679,International Terminal Main Hall,concourse
51131288670,37.612777,-122.361112,1730008749,RUNWAY 10R/28L,custom
```

The `pip` tool is designed to operate with any Flickr API method that returns a **standard photo response** `https://code.flickr.net/2008/08/19/standard-photos-response-apis-for-civilized-age/` so it will also work fetching and reverse-geocoding all the openly-licensed and geotagged photos from the **airports-sfo** `https://www.flickr.com/groups/airports-sfo/pool/` Flickr group:

```
$> go run -mod vendor cmd/pip/main.go \
        -client-uri 'oauth1://?consumer_key=...&consumer_secret=...' \
        -param method=flickr.photos.search \
        -param group_id=95693046@N00 \
        -param has_geo=1 \
        -param extras=geo \
        -param license=1,2,3,4,5,6,7,8,9,10 \
        > airports.csv
```

I've uploaded a copy of **the airports.csv file** `airports.csv` if you want to see all the results but in the interest of brevity here's the unique list of places those photos depict when we query SFO Museum's own **architecture data** `https://github.com/sfomuseum-data/sfomuseum-data-architecture` :

```
$> cat airports-sfo.csv | awk -F ',' '{ print $5 }' | sort | uniq
Boarding Area D
Boarding Area F
Boarding Area G
Central Parking Garage
```

```
D-13 Kids' Spot
Garage A
Garage A AirTrain Station
Garage G
Garage G AirTrain Station
International Terminal
International Terminal Connector
International Terminal Departures (Post-Security)
International Terminal Main Hall
Null Terminal
RUNWAY 01L/19R
RUNWAY 10R/28L
SFO Bart Station
SFO Terminal Complex
Taxiway A/F1
Taxiway D
Taxiway G
Taxiway Q
Terminal 2
Terminal 2 Air Train Station
Terminal 2 Main Hall
Terminal Two Arrivals
```

Null

Terminal `https://millsfield.sfomuseum.org/id/1729828959/` *, if you're wondering, is a liminal space located at the intersection of the two runways where we place things that we know are "at" SFO but we're still not exactly sure where.*

Negative: San Francisco International Airport (SFO), aerial. Negative. Transfer, SFO Museum Collection. ***2011.032.1446***

As the **Reverse-Geocoding in Time at SFO Museum** https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/ blog post demonstrated there are actually many other "places" those photos might "be" depending on when they were taken but, for now, the tools in the **go-flickr-pip** https://github.com/sfomuseum/go-flickr-pip package are designed to limit results to places that are considered "current".

In the **Surface Areas – Photos and Depictions on the Mills Field Website** https://millsfield.sfomuseum.org/blog/2019/01/02/surface/ blog post I wrote about how SFO Museum has started to **import openly-licensed Flickr photos** https://millsfield.sfomuseum.org/images/flickr/ of airports and airplanes relevent to SFO Museum and said:

> Some of those 100,000 photos aren't relevant or germane to our collection (and some of them aren't even at airports because geotagging photos is hard) so the immediate concern will be developing tools and interfaces to let the museum work through the queue.

The `go-flickr-pip` tool is mostly a demonstration of what's possible but it also serves as kind of "training-wheels" for how we might automate and improve the processes that we use for importing data from Flickr or any other third-party data source. It's also very relevant to our on-going work building **tools for geotagging our own collection objects** https://millsfield.sfomuseum.org/blog/tags/geotagging .

These tools and the data that we used to do this work are not specific to Flickr or SFO Museum either. The other day we released the **smk-open-to-wof** https://github.com/sfomuseum/smk-open-to-wof package

which contains "tools for building a mapping of National Gallery of Denmark (SMK) collection objects and their latitude and longitude coordinates to corresponding **Who's On First** `https://whosonfirst.org/what/` IDs".

In 1954, SAS inaugurated the first service between California and Copenhagen, Denmark. ***Twitter, August 25, 2015***

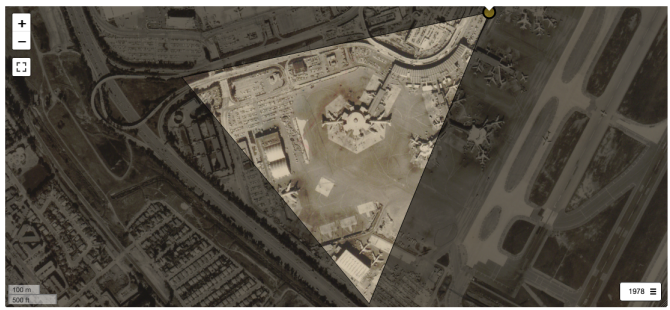
It uses the same `wof-sqlite-index-features` and `go-whosonfirst-spatial-www-sqlite` tools that we used to reverse-geocode Flickr photos at SFO but with **location data for the country of Denmark** `https://github.com/whosonfirst-data/whosonfirst-data-admin-dk` and **geotagged collections data from the National Gallery of Denmark** `https://danmarkskort.open.smk.dk/` . For example here are a few of the collection objects that are located in WOF ID **102544207** `https://spelunker.whosonfirst.org/id/102544207`

which is the **Copenhagen Airport** `https://millsfield.sfomuseum.org/airports/102544207/flights/` , an airport relevant to the history and interests of SFO Museum:

```
> git grep 102544207 smk-open-wof.csv
smk-open-wof.csv:1684,55.63568419562137,12.640782935717771,102544207,85907903,Copenhagen
smk-open-wof.csv:1680,55.62584400000001,12.6227171,102544207,85907903,Copenhagen Airport,
smk-open-wof.csv:2314,55.59660361967791,12.634810984130844,102544207,85907903,Copenhagen
smk-open-wof.csv:2316,55.594276030589086,12.634510576721176,102544207,85907903,Copenhagen
smk-open-wof.csv:2301,55.62714440000001,12.6236595,102544207,85907903,Copenhagen Airport,
smk-open-wof.csv:3114,55.61155381950298,12.669836623767088,102544207,85907903,Copenhagen
smk-open-wof.csv:3173,55.61482775161416,12.66678963432617,102544207,85907903,Copenhagen A
smk-open-wof.csv:3177,55.613474191461094,12.668248756030271,102544207,85907903,Copenhagen
```

Coming full circle, the description for the photo of **South Terminal** `https://millsfield.sfomuseum.org/search/?q=South%20Terminal&placetype=terminal` , above, reads:

> 35mm photographic color negative depicting exterior view taken from upper level of Central Terminal (now Terminal 2 / T2) facing southwest of South Terminal (now Terminal 1 / T1) gates with airplanes parked, roadway and parking lot; Millbrae and Burlingame visible in background.



And it turns out that it's **one of the objects we've**

geotagged `https://millsfield.sfomuseum.org/objects/1511949289/#1978/16/37.6122/-122.3889` already as part of **Geotagging at SFO Museum** `https://millsfield.sfomuseum.org/blog/tags/geotagging` blog posts. I mentioned earlier that the `go-flickr-pip` tool is limited to returning records that are considered "current". In future blog posts I'll talk about how we plan to use these reverse-geocoding tools to help locate objects in time and how that work will be integrated with our tools for geotagging collections objects.

2021-04-26

everything is everything else

Stil life (2021)

Stil life (2021)

2021-04-27

conversations with math

Any sufficiently advanced correlation is...

Any sufficiently advanced correlation is indistinguishable from causation

Here is a lightly-annotated collection of articles and papers, on or about the question of inference, that have struck a chord recently. And some drawings and pictures thrown in for good measure.

1. ***Data ordering attacks*** `https://www.lightbluetouchpaper.org/2021/04/23/data-ordering-attacks/` *, Light Blue Touchpaper*

> Most deep neural networks are trained by stochastic gradient descent. Now "stochastic" is a fancy Greek word for "random"; it means that the training data are fed into the model in random order. So what happens if the bad guys can cause the order to be not random? You guessed it – all bets are off. Suppose for example a company or a country wanted to have a credit-scoring system that's secretly sexist, but still be able to pretend that its training was actually fair. Well, they could assemble a set of financial data that was representative of the whole population, but start the model's training on ten rich men and ten poor women drawn from that set – then let initialisation bias do the rest of the work. Does this generalise? Indeed it does. Previously, people had assumed that in order to poison a model or introduce backdoors, you needed to add adversarial samples to the training data. Our latest paper shows that's not necessary at all. If an adversary can manipulate the order in which batches of training data are presented to the model, they can undermine both its integrity (by poisoning it) and its availability (by causing training to be less effective, or take longer). This is quite general across models that use stochastic gradient descent.

1. ***Monster Mash: A Sketch-Based Tool for Casual 3D Modeling and Animation*** `https://ai.googleblog.com/2021/04/monster-mash-sketch-based-tool-for.html` *, Google AI Blog*

This was interesting to me for two reasons: 1) It is the kind of

generic toolkit that could/should be applied to any number of interactive contexts in any number of museums 2) It highlights the lack of capacity across the sector to be able to take something like this and adapt it to its own specific needs. I expect that, inside of five years, at least one large institution will blow 6 or 7 figures hiring an external firm to implement this as a one-off which is... a missed opportunity, really.

1. ***An update on the UMN affair*** `https://lwn.net/SubscriberLink/854645/334317047842b6c3/` *, Linux Weekly News*

One final lesson that one might be tempted to take is that the kernel is running a terrible risk of malicious patches inserted by actors with rather more skill and resources than the UMN researchers have shown. That could be, but the simple truth of the matter is that regular kernel developers continue to insert bugs at such a rate that there should be little need for malicious actors to add more. The 5.11 kernel, released in February, has accumulated 2,281 fixes in stable updates through 5.11.17. If one makes the (overly simplistic) assumption that each fix corrects one original 5.11 patch, then 16% of the patches that went into 5.11 have turned out (so far) to be buggy. That is not much better than the rate for the UMN patches.So perhaps that's the real lesson to take from this whole experience: the speed of the kernel process is one of its best attributes, and we all depend on it to get features as quickly as possible. But that pace may be incompatible with serious patch review and low numbers of bugs overall. For a while, we might see things slow down a little bit as maintainers feel the need to more closely scrutinize changes, especially those coming from new developers. But if we cannot institutionalize a more careful process, we will continue to see a lot of bugs and it will not really matter whether they were inserted intentionally or not.

1. ***How image search works at Dropbox*** `https://dropbox.tech/machine-learning/how-image-search-works-at-dropbox` , *Dropbox*

2. ***Generative Models as a Robust Alternative for Image***

Classification: Progress and Challenges https://www2.eecs.berkeley.edu/Pubs/TechRpts/2021/EECS-2021-47.html , *An Ju*

3. ***ALIGN: Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision*** https://ai.googleblog.com/2021/05/align-scaling-up-visual-and-vision.html , *Google AI Blog*

All three papers are a reminder that almost all machine learning work is predicated on disproving the adage that past performance is not an indicator of future success, which... well, **you know** https://www.aaronland.info/weblog/2019/12/05/iconography/#fantastic .

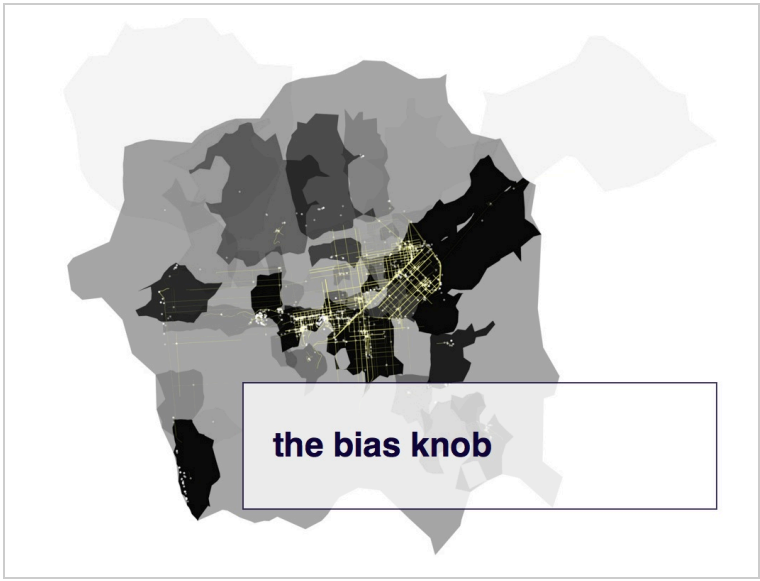
1. ***Enhancing Photorealism Enhancement*** https://intel-isl.github.io/PhotorealismEnhancement/ , *Intel Labs*

This is very impressive work but the phrase "It greens the parched grass and hills in Grand Theft Auto's (GTA) California" sort of sums it up. It's clear that GTA is a MacGuffin for the underlying work but it betrays a disservice to the material quality of the GTA's aesthetics and the relentless focus on "style transfers" suggests that it's just pumpkin spice lattes all the way down...

1. ***A conversation with Harvard GSD's Charles Waldheim about Ed Ruscha's archive at the Getty*** https://hi.stamen.com/introduction-

378564373c7c , *Stamen*

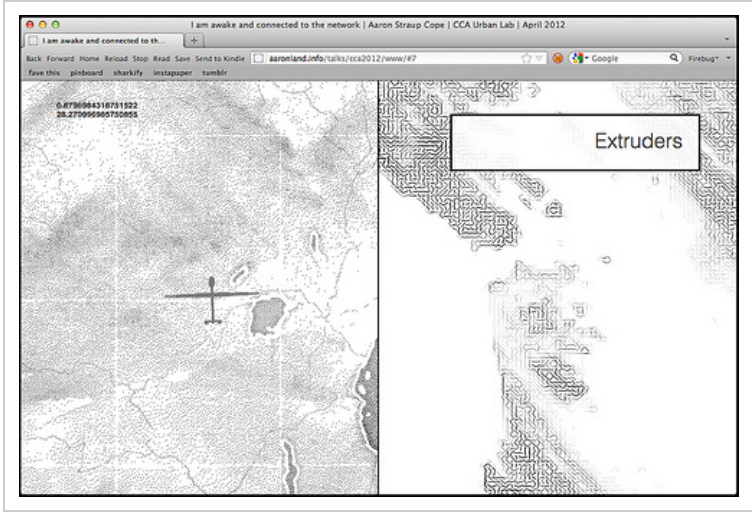
During the time that I was at Stamen we had started to talk about a "bias knob" for data visualizations. This was largely a rhetorical device since no one was quite sure what the technology to make such a thing possible would be. It is nice to see the use of machine-learning in the work that GSD is doing framed in similar terms, that it is a fast and efficient tool for applying pressures to different parts of a dataset and seeing what happens rather than a magic "veil" of truth. These are some of the things I wrote about this at the time:



One of the things I like about this image is that you can clearly see that Fort Funston, the place where H. goes to walk her dogs, is part of *her* San Francisco. You can see this not because of the roads on the beach; there are none. You can see this because the photos cause the neighbourhood around the beach to materialize.

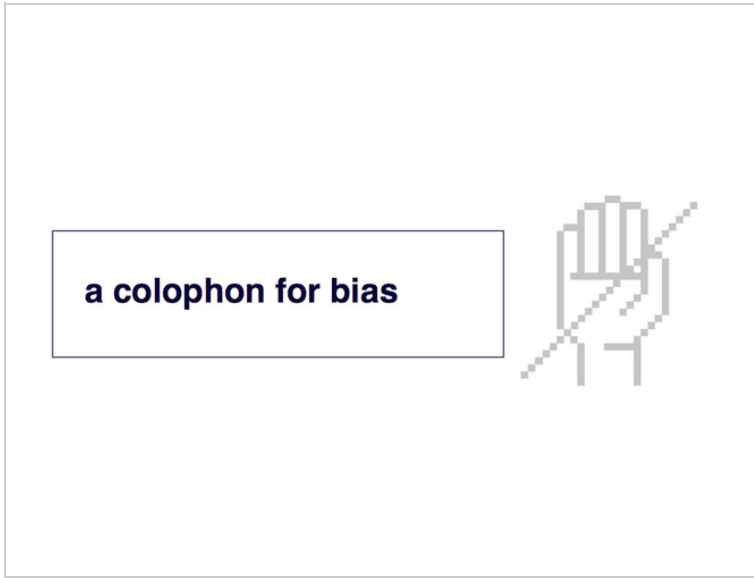
In a lot of generative art works, and in game design, there is the idea of a "physics knob" that you can use to adjust the relationships between different objects and the environment in which they exist. I've always tried to imagine what it would mean to have a "bias knob" that you could use to affect how cause and effect, or in this case maps, are displayed.

we need / MOAR dragons `https://www.aaronland.info/weblog/2010/11/14/andthen/#dragons`



In the past I've talked about producing and treating maps the same way you might work with a bolt of fabric but maybe it's also like making pasta where the data (and the choice of data) acts as a kind of extruder shaping the noodles. I don't really want to get lost in bad kitchen metaphors so I'm going to stop there and instead leave you with the image of the raw data acting as a kind of screen through which the squishy mass of history, of time and place, is passed to create a map.

I am awake and connected to the network `https://www.aaronland.info/weblog/2012/04/02/haystack/#connected`



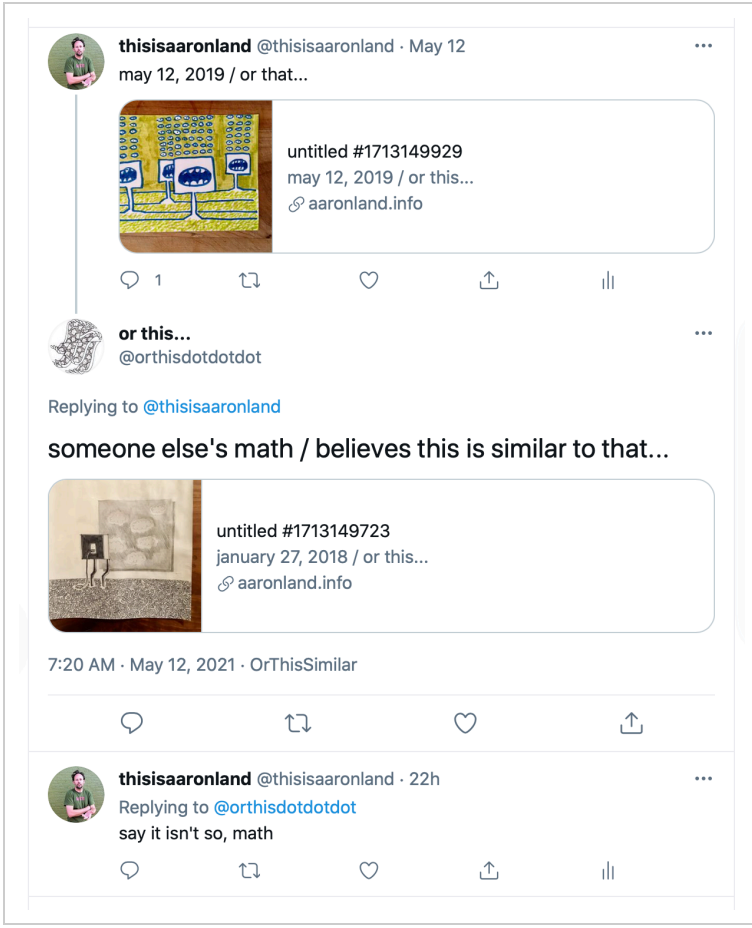
What search-by-color and other algorithmic cataloging points to is the need to develop an iconography, or a colophon, to indicate machine bias. To design and create language and conventions that convey the properties of the "extruder" that a dataset has been shaped by.

a colophon for bias https://aaronland.info/weblog/2014/10/27/things/#colophon

1. ***Image "Cloaking" for Personal Privacy*** `https://sandlab.cs.uchicago.edu/fawkes` *, SAND Lab*

The SAND Lab at University of Chicago has developed Fawkes1, an algorithm and software tool (running locally on your computer) that gives individuals the ability to limit how unknown third parties can track them by building facial recognition models out of their publicly available photos. At a high level, Fawkes "poisons" models that try to learn what you look like, by putting hidden changes into your photos, and using them as Trojan horses to deliver that poison to any facial recognition models of you. Fawkes takes your personal images and makes tiny, pixel-level changes that are invisible to the human eye, in a process we call image cloaking. You can then use these "cloaked" photos as you normally would, sharing them on social media, sending them to friends, printing them or displaying them on digital devices, the same way you would any other photo. The difference, however, is that if and when someone tries to use these photos to build a facial recognition model, "cloaked" images will teach the model an highly distorted version of what makes you look like you. The cloak effect is not easily detectable by humans or machines and will not cause errors in model training. However, when someone tries to identify you by presenting an unaltered, "uncloaked" image of you (e.g. a photo taken in public) to the model, the model will fail to recognize you.

Which bookends nicely with the paper on data ordering attacks.

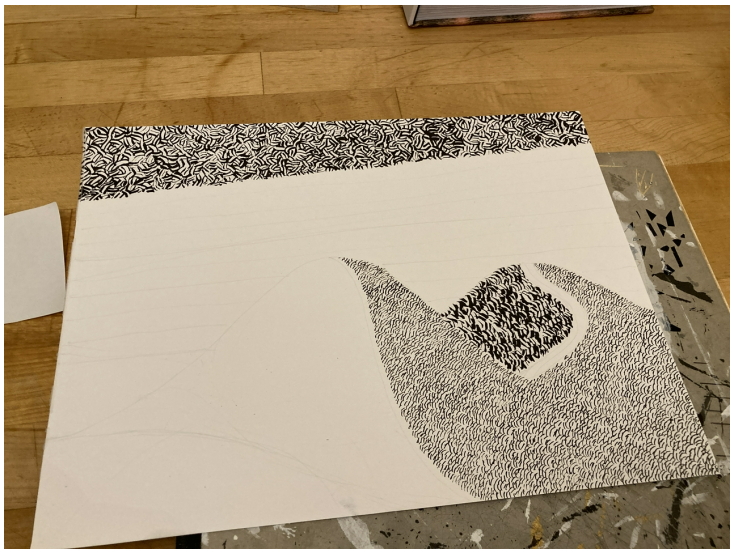

thisisaaronland @thisisaaronland · May 12
may 12, 2019 / or that...
untitled #1713149929
may 12, 2019 / or this...
aaronland.info
1
or this...
@orthisdotdotdot
Replying to @thisisaaronland
someone else's math / believes this is similar to that...
untitled #1713149723
january 27, 2018 / or this...
aaronland.info
7:20 AM · May 12, 2021 · OrThisSimilar
thisisaaronland @thisisaaronland · 22h
Replying to @orthisdotdotdot
say it isn't so, math

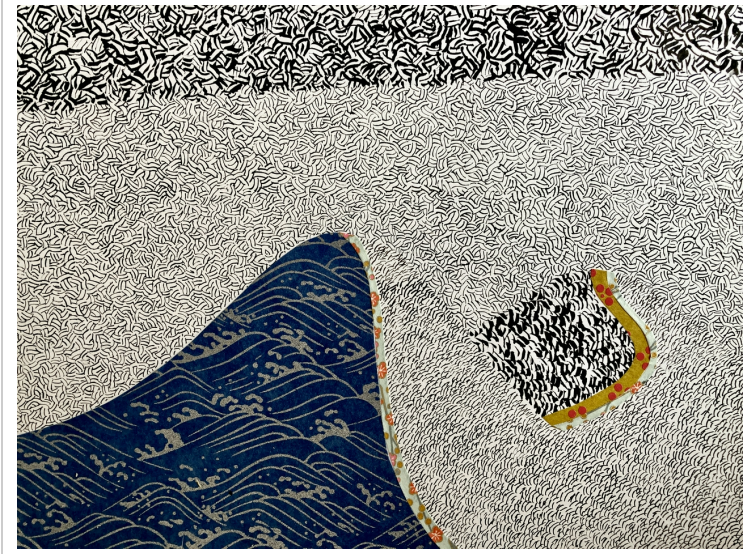
2021-05-17

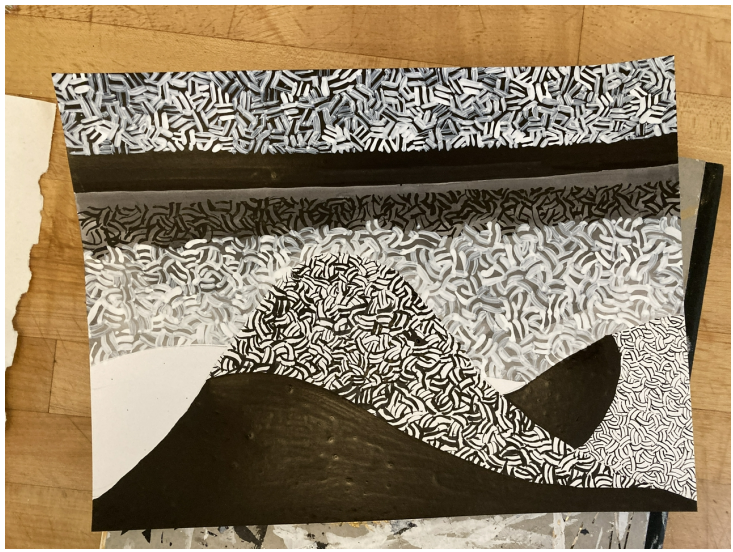
amplify like no one is watching

Two views from the 80 East (2021)

Two views from the 80 East (2021)







Somewhere after the Carquinez Bridge, and Northeast of the **580 East** `https://aaronland.info/weblog/2021/04/09/waves/#580` .

2021-05-19

a trauma-hammer experience

Four works with chiyogami (2021)

Four works with chiyogami (2021)

Three of the four works **are available for sale** `https://or-this.myshopify.com/collections/all/chiyogami?sort_by=created-descending` .

2021-05-29

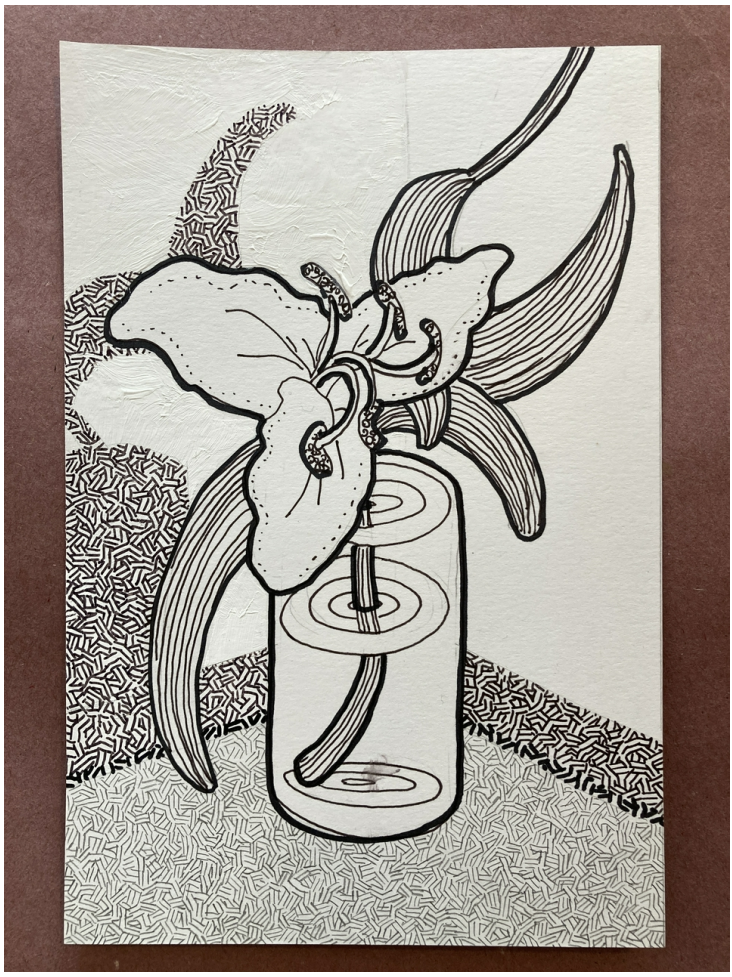
Artist McArtistface

Big Fabric Lily (2020-21)

Big Fabric Lily (2020-21)

Here is a thing I have been working on.

This is the first large textile work derived from a drawing since the **the three-panel hanging of Mori Point** https://www.aaronland.info/weblog/2020/09/22/allusion/#mori-point last year. That piece was digitally printed which allowed each panel to be **nine feet tall** https://aaronland.info/orthis/172/957/678/9/ but, while unique in its own right, lacks the luster of hand-printed inks.



As with the **digital prints** https://or-this.myshopify.com/collections/prints this piece is derived from a vectorization of a drawing I made last year, meaning it is **the same yet different** https://aaronland.info/weblog/2020/10/27/uniformity/#same .

The drawing itself is only 6 x 9 inches made with ink and pencil and gesso on paper. The fabric prints are 43 x 65 inches.

Shortly after I scanned the drawing I touched it with wet fingers. This happens more than I'd like to admit.

The prints were made with **Anne Kirk** https://www.annekirktextiles.com/ .

Anne was kind and patient and a pleasure to work with as I negotiated a whole new **matter battle** `http://www.matterbattle.com/` of production.

There is an edition of five prints, signed and numbered, **available for sale** `https://or-this.myshopify.com/collections/textiles/products/lily-2020-21` .

The prints are not signed "Artist McArtistface". They are printed on a thin muslin so this was a test to see whether a signature on the hem would be visible when folded over. It remains a tempting alter-ego.

2021-06-03

style transfers for the deluge

Two views of Mori Point (2021)

Two views of Mori Point (2021)

One of the two works **is available for sale** https://or-this.myshopify.com/collections/frontpage/products/mori-point-chiyogami-2021 .

2021-06-08

things I have written about elsewhere #20210616

Geotagging at SFO Museum: Client-side EXIF, Web...

Geotagging at SFO Museum: Client-side EXIF, Web Components, Tilepacks, AppRunner and Live Demos

This was **originally published on the Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2021/06/16/geotagging/` , in June, 2021.

Negative: San Francisco Airport, meeting with National Security Resources Board. Negative. Transfer, SFO Museum Collection. ***2011.032.0179***.

Here is an update on a number of improvements we've implemented since the **last blog post about the geotagging project** `https://millsfield.sfomuseum.org/blog/tags/geotagging` . Specifically:

1. *Client-side EXIF encoding of geotagging coordinates in downloadable images*.

2. *Support for drag-and-drop and file-browser-based image selection*.

3. *Bundled and locally served Nextzen map tiles*.

4. *Support for running the geotagging application with bundled databases inside of a Docker container, and deployed as an AWS AppRunner instance*.

5. ***A live demonstration application*** `https://millsfield.sfomuseum.org/demo/geotag/` .

Client-side EXIF

`https://millsfield.sfomuseum.org/objects/1511943955/`

Negative: San Francisco Airport, Flying Tiger Line Curtiss C-46 Commando. Negative. Transfer, SFO Museum Collection. ***2011.032.0327*** `https://millsfield.sfomuseum.org/objects/1511943955/` .

It is now possible to configure the web application to encode geotagging information for an image in to a *new image* that you can download to your computer. All of this happens in the browser so there is no need for a server-side component to encode geotagging information. This builds on the work we've been doing to develop **a WebAssembly-based EXIF update tool** `https://millsfield.sfomuseum.org/blog/tags/wasm` .

At the moment only the geotagging center point is recorded as EXIF `GPSLatitude` *and* `GPSLongitude` *properties. Field of view information is not included yet while I figure out how and where the best place to encode that data within the constraints of the EXIF format.*

Web Components

`https://millsfield.sfomuseum.org/objects/1729563855/`

Photograph: Transcontinental & Western Air (TWA), Douglas DC-1. Photograph. Gift of Carla B. Bos in memory of Abraham Bos, SFO Museum Collection. ***2000.069.016*** `Photograph. Gift of Carla B. Bos in memory of Abraham Bos, SFO Museum Collection. 2000.069.016. .`

We have integrated GitHub's **`file-attachment` WebComponent** `https://github.com/github/file-attachment-element` with the application. This provides user-interface elements that allow a person to drag-and-drop, or select from their computer, an image they want to geotag.

WebComponents `https://developer.mozilla.org/en-US/docs/Web/Web_Components` are a new-ish browser technology, described as:

> ...a suite of different technologies allowing you to create reusable custom elements — with their functionality encapsulated away from the rest of your code — and utilize them in your web apps.

I like this because I hear echoes of something I said **in the very first blog post** `https://millsfield.sfomuseum.org/blog/2020/04/23/geotagging/` about geotagging images:

> ...we would be better served as a sector to focus on shared tools that automate what can happen to data and making it both easy to ingest and export in the process. The cultural heritage sector needs as many of these small, focused, tools as it can produce. It needs them in the long-term to finally reach the goal of a common infrastructure that can be employed sector-wide.

It is possible to imagine a world where there are too many WebComponents, of variable quality and in some cases questionable motives. That's a risk we should be conscious of but there is also the potential for well-designed, well-engineered and genuinely useful pieces of functionality, like the `file-attachment` component demonstrates, that we can share and build upon.

At the moment, only individual images are handled by the geotagging application. The `file-attachment` element supports loading multiple images but the geotagging application does not yet.

Tilepacks

Photograph: KLM (Royal Dutch Airlines), TWA (Trans World Airlines), San Francisco International Airport (SFO), Julian Veovich. Photograph. Gift of Julian Veovich, SFO Museum Collection. ***2013.126.023****.*

In the **last blog post** `https://millsfield.sfomuseum.org/blog/2021/05/03/geotagging/` I talked about using the **Protomaps** `https://protomaps.com/` mapping libraries and databases for serving map tiles without the need for a separate database or hosting infrastructure. I am pleased to announced that we've enabled similar functionality for **Nextzen** `https://nextzen.org/` map tiles. Nextzen map tiles are rendered using the **Tangram.js** `https://github.com/tangrams/tangram` library which has been around longer than Protomaps and has better support for things like labels. Each is great in their own right and ensuring that both are supported in the geotagging application is even better.

A local database of Nextzen tiles can be created using the tools in the Tilezen **go-tilepacks** `https://github.com/tilezen/go-tilepacks` package. For example this is how we created a database of map tiles for

SFO:

```
$> cd go-tilepacks
$> make tools

$> ./bin/build \
        -dsn /usr/local/data/sfo-tiles.db \
        -zooms 0-16
        -bounds '37.604481,-122.405228,37.645194,-122.355044' \
        -url-template 'https://tile.nextzen.org/tilezen/vector/v1/512/all/{z}/{x}/{y}.mvt
```

And then to use these tiles in our geotagging application we include the following flags when we start the application:

```
$> cd go-www-geotag

$> bin/server \
        -enable-tilezen-tilepacks true \
        -tilezen-tilepack-path /usr/local/data/sfo-tiles.db \
        -nextzen-tile-url /tilezen/vector/v1/512/all/{z}/{x}/{y}.mvt

        ...other geotagging application configuration flags
```

The map tiles themselves are harvested from the Nextzen website and stored in a **SQLite** `https://www.sqlite.org/` (or **MBTiles** `https://github.com/mapbox/mbtiles-spec`) database that is then used by the geotagging application itself to serve its own map tiles. That means there one less external service to depend on long-term for running a geotagging application. It's not necessarily an approach that will scale to an application which has to support a large geographic extent but for small and bounded use cases it's nearly perfect.

AppRunner

Negative: San Francisco International Airport (SFO), SFO Helicopter Airlines. Negative. Transfer, SFO Museum Collection. ***2011.032.1289****.*

Recently, Amazon Web Services (AWS) announced their new **AppRunner** `https://docs.aws.amazon.com/apprunner/latest/dg/getting-started.html` service. AppRunner is the distillation, and the simplification, of a number of existing AWS products and networking components for hosting web applications that are packaged as (Docker) containers.

Put another way, AppRunner is "fewer buttons to press" to deploy a web application for a lower monthly fee than before. AppRunner ***doesn't have all the features*** `https://github.com/aws/apprunner-roadmap/issues` *that other AWS services do but it many cases that doesn't matter.*

A "container" is a standardized way of packaging and running web applications independent of the specific implementation details of a hosting provider (like AWS). In the **Using the Placeholder Geocoder at SFO**

Museum `https://millsfield.sfomuseum.org/blog/2019/11/04/placeholder/` blog post I wrote that:

> Docker is both a service and an application for running virtual machines which are referred to as "containers". A container is basically a freeze-dried virtual computer that contain only that which is absolutely necessary to run a pre-defined set of applications. Tools like Docker take care of "thawing" (and re-freeze-drying) containers and running the software they contain. It's a pretty clever abstraction and managing the operation of containers, as a service, has been adopted by all the different cloud providers.

I like to think of this as layers, or separations, of concern:

1. *My computer, the source code for the* `go-www-geotag` *application.*

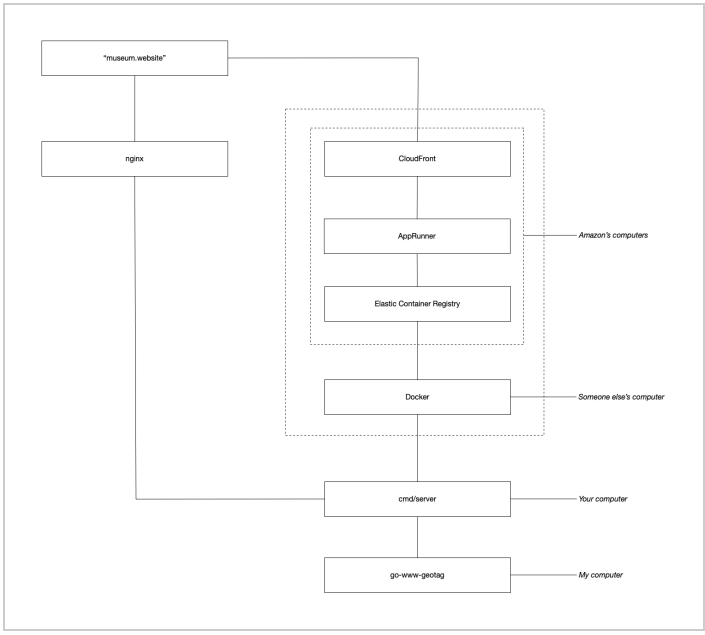
2. *Your computer, the compiled* `go-www-geotag` *application running locally in a web browser.*

3. *Someone else's computer, the* `go-www-geotag` *application running inside a virtual computing environment (aka a "container").*

4. *Amazon's computer, the* `go-www-geotag` *container running inside Amazon's "cloud".*

We've made sure that the `go-www-geotag` application doesn't depend on AWS, Docker or containerized computing environments to run. Containers are all the rage these days but that may not always be true or practical. The `go-www-geotag` application can take advantage of the benefits that each of these technologies provide but you can just as easily

run the application on its own or behind a traditional web server like **Apache** `https://httpd.apache.org/` or **Nginx** `https://nginx.org/` if you want, or more importantly, need to.



This image is quite large so it might be easier to read **by looking at the full-sized original**.

The `go-www-geotag` package is bundled with a **Dockerfile** `https://github.com/sfomuseum/go-www-geotag#docker` for building a containerized version of the application. It has been updated to look for any Protomaps or MBTiles database files you put in a local `data` folder and copies them in to the final container. This means that you can deploy an entirely self-contained application, inclusive of map tiles and spatial databases, as a single "container" resource that you pass simply along to AWS, or another cloud provider, to host.

This is an approach we've been actively investigating, starting with the **Reverse-Geocoding in Time at SFO Museum** `https://millsfield.sfomuseum.org/blog/2021/03/26/spatial/` blog post. Conceptually the reverse-geocoding demonstration applications described in that blog post and the **geotagging demonstration** `https://millsfield.sfomuseum.org/demo/geotag/` , described below, are the same. The big difference is that the latter is easier to configure and deploy than the former.

Live demos!

Negative: San Francisco International Airport (SFO), fire fighting demonstration. Negative. Transfer, SFO Museum Collection. ***2011.032.0618***.

There is a live demonstration application of everything described in this blog post:

https://millsfield.sfomuseum.org/demo/geotag/ `https://millsfield.sfomuseum.org/demo/geotag/`

While **this particular instantiation** `https://millsfield.sfomuseum.org/demo/geotag/` of the geotagging application is scoped to the physical boundaries of SFO **the code itself** `https://github.com/sfomuseum/go-www-geotag` is not specific to the airport or SFO Museum. It can be used, in combination with custom databases of map tiles and geographic data, with any image on your computer. This code will continue to be the common infrastructure that we build a SFO Museum application, specific to our needs, on top of but it has been designed with general use in mind.

If you'd like to contribute there are a number of **user-interface issues** `https://github.com/sfomuseum/go-www-geotag/issues?q=is%3Aissue+is%3Aopen+label%3Aui` and **user-experience issues** `https://github.com/sfomuseum/go-www-geotag/issues?q=is%3Aissue+is%3Aopen+label%3Aux` that we would welcome comments and suggestions and help with.

https://github.com/sfomuseum/go-www-geotag `https://github.com/sfomuseum/go-www-geotag`

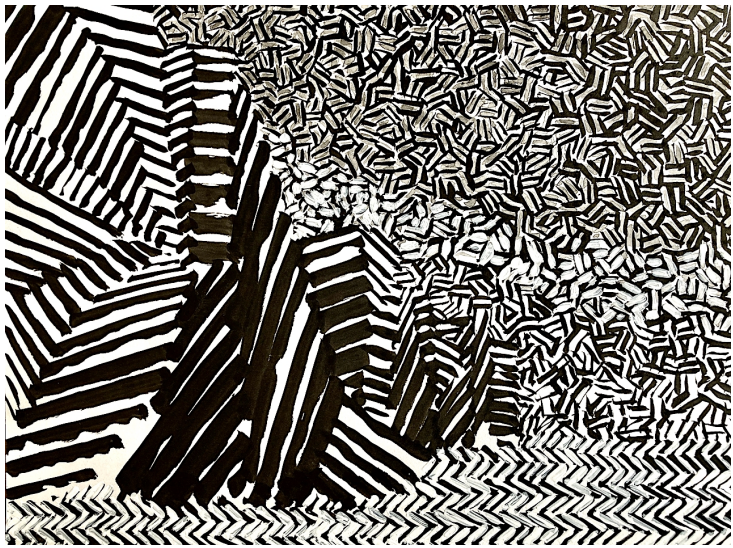
2021-06-16

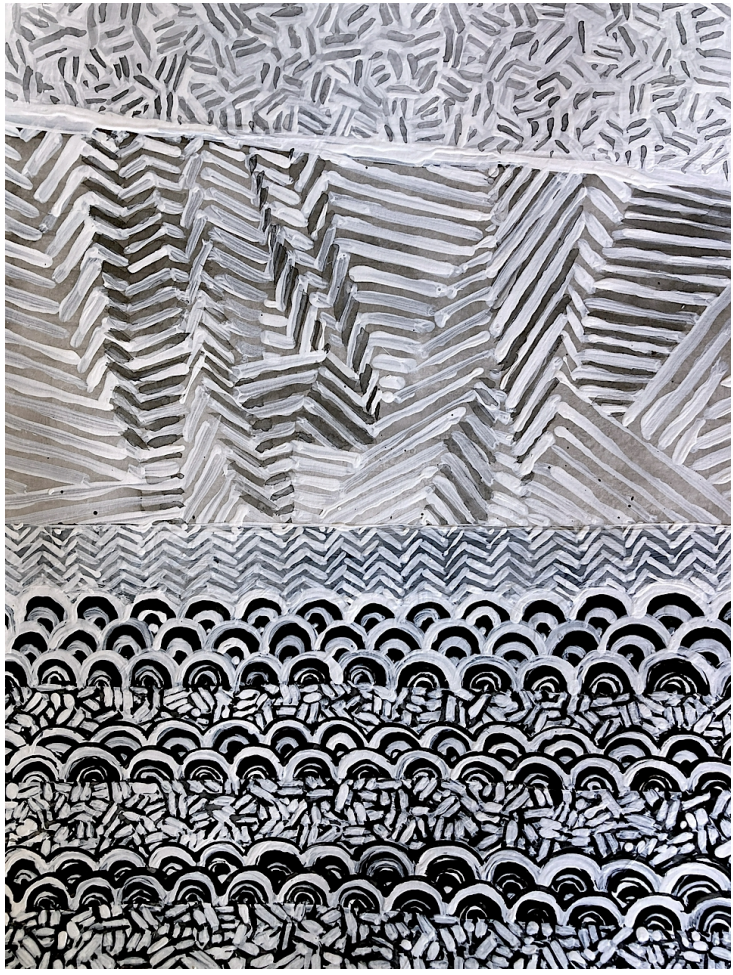
machine-learning or bucket list: can you tell the difference?

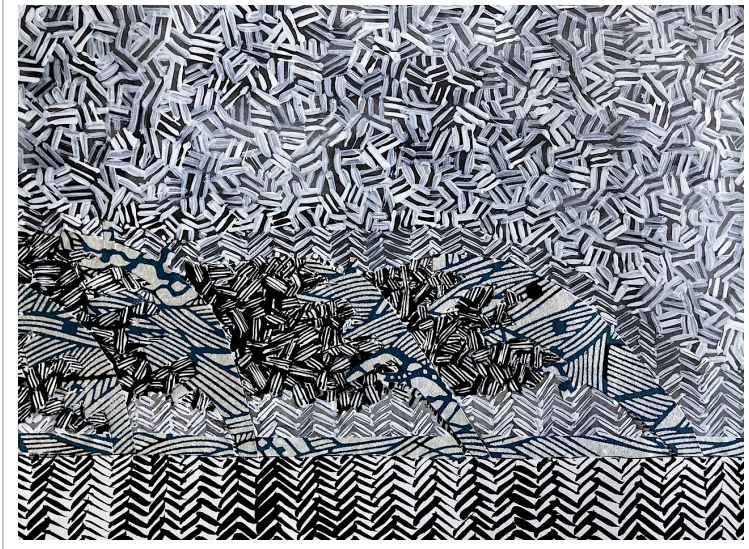
Eight views of Ocean Beach (2021)

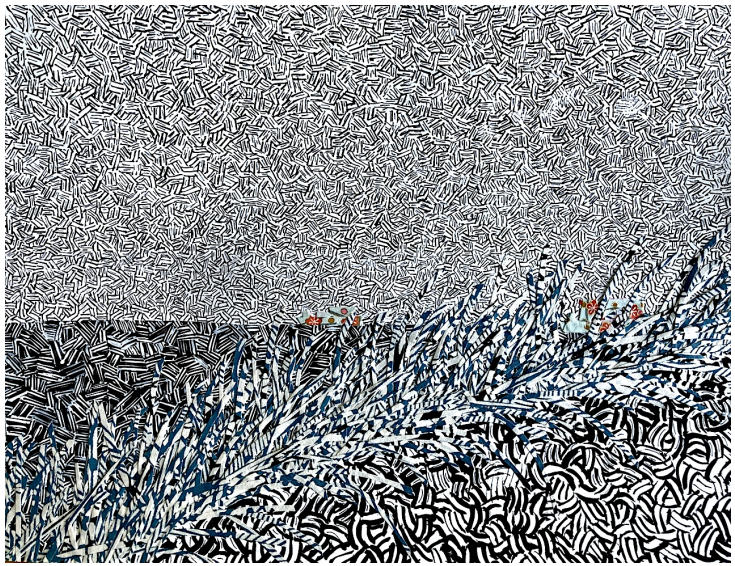
Eight views of Ocean Beach (2021)

This one is about a quarter of the size of the others.









I've made some of the works **available for sale** `https://or-this.myshopify.com/collections/ocean-beach` .

2021-06-21

urgency or it didn't happen

Knives (2021)

Knives (2021)

This work is available **for sale** `https://or-this.myshopify.com/products/knives-chiyogami-2021` .

2021-06-29

machine learning the past to gaslight the future

Five works on frozen fish cake palettes (2021)

Five works on frozen fish cake palettes (2021)

PELICAN
APPRECIATION
CLUB

SOCIÉTÉ DES POISSONS
IMAGINAIRES

415
514

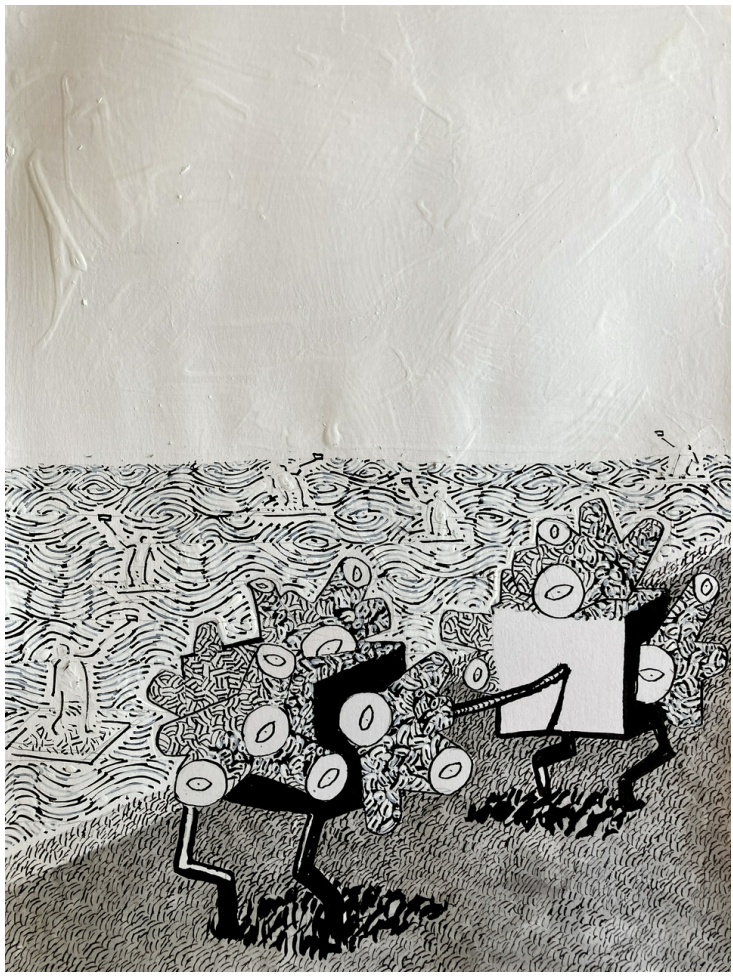
I've been making soup that calls for frozen fish cakes recently. Most seem to be sold on top of cute little wooden palettes. While charming, they also feel like an exercise in wasteful packaging so it's nice to find a secondary use for them.

2021-07-12

understanding through adrenaline

Six works with the bollard-things, and some...

Six works with the bollard-things, and some selfie-sticks (2021)



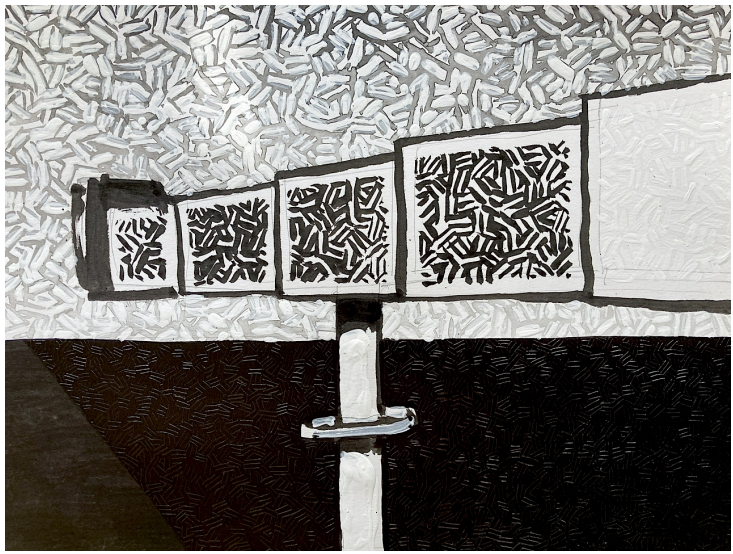
I have made some of the pieces **available for sale** `https://or-this.myshopify.com/collections/six-works-with-the-bollard-things-and-some-selfie-sticks-2021` .

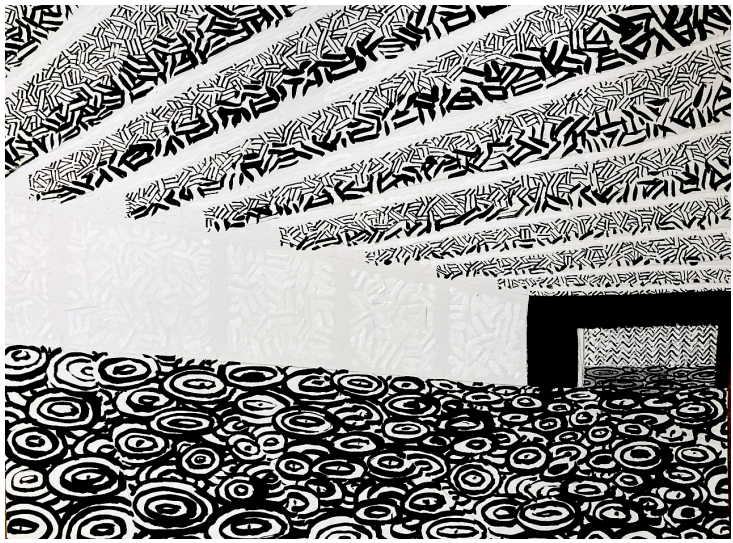
2021-07-26

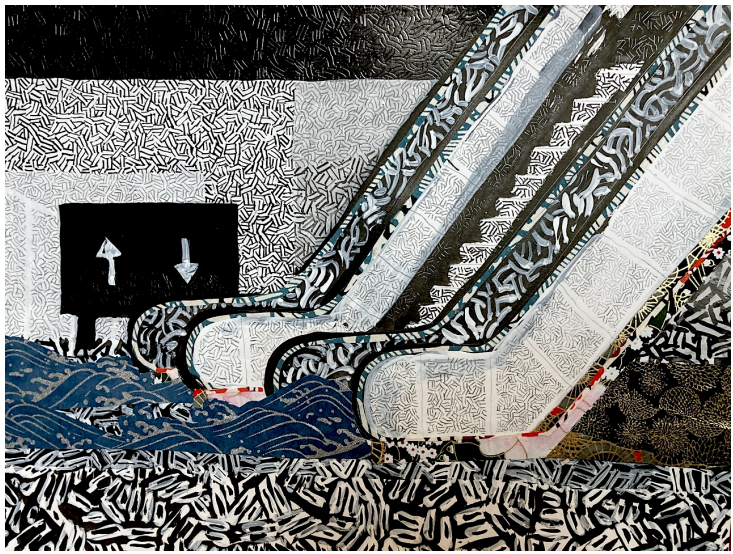
generative adversarial modernism

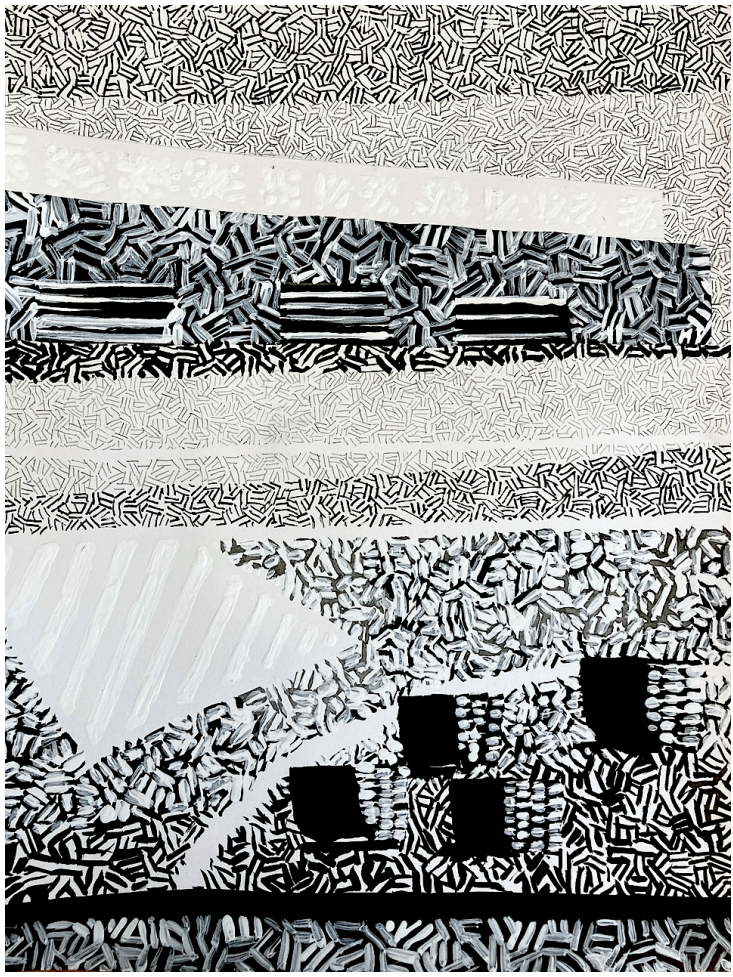
Five works "of" airports (2021)

Five works "of" airports (2021)









As I write this, I like two of these works. I am on the fence about another pair and we'll see what **the space of time** `https://aaronland.info/weblog/2020/06/08/covert/#date` does to the other one. I spend **a lot of time at SFO** `https://millsfield.sfomuseum.org/blog/` so obviously some of these are influenced by what I see there. Despite that and despite the recent trend of airports developing signature features and recognizable attractions it is still the case that most elements of most airports remain identical and largely interchangeable. This is not a novel observation but it is interesting to consider in light of the growing number of machine-generated "is this real or not?" -style datasets that are being produced these days.

2021-08-31

generative dependencies

go-marc

go-marc

Flight information packet: American Airlines. Collection of SFO Museum. **2013.088.002 a l**.

A couple of weeks ago I attended a presentation about the **navPlace extension to the IIIF APIs** `https://preview.iiif.io/api/navplace_extension/api/extension/navplace/` . The extension defines the semantics for appending location information to IIIF resources, typically old historical maps, as

GeoJSON FeatureCollection

elements `https://datatracker.ietf.org/doc/html/rfc7946` . At one point during the presentation someone pointed out that there are a lot of institutions with maps lacking geographic information or that only have geographic information stored in MARC (or **machine-readable cataloging** `https://en.wikipedia.org/wiki/MARC_standards`) records. Usually that information is stored in **034 - Coded Cartographic Mathematical**

Data `https://www.loc.gov/marc/bibliographic/bd034.html` fields that look like this:

```
1#$aa$b80000$dW0825500$eW0822000$fN0273000$gN026500
```

That's the MARC 034 encoding for a bounding box containing the area spanning

Bradenton `https://spelunker.whosonfirst.org/id/85932761/` and **Venice** `https://spelunker.whosonfirst.org/id/85932705/` along Florida's Gulf cost and I was reminded that I had actually written code for working with these data.

In 2017 I spent a few days with the **DX Labs** `https://dxlab.sl.nsw.gov.au/` crew at the **State Library of New South Wales** `sl.nsw.gov.au` in Sydney. **Mapzen** `https://www.wired.com/story/mapzen-shuts-down/` was still a going concern then and I was actively working on the **Who's On First** `https://www.whosonfirat.org/` gazatteer project. The Library has an extensive collection of maps of the greater Sydney area. Many of these maps aren't geographically accurate, taking artistic license in their depiction of space, but over the years have been assigned a geographic extent (a bounding box) based on what is visible in the map.

Because I was working on a gazetteer at the time my first thought was that it would be interesting to find all the places, (the neighbourhoods, microhood, localities) that intersected the extent of any given map. That

would allow the library to assign persistent identifiers for specific places and enable spatial lookups of their map collection without necessarily having to do perform spatial operations or manage a database of spatial information necessary to perform those queries.

Traveler information brochure: Air France. Collection of SFO Museum. ***2011.001.069***

In order to do this it was first necessary to account for the fact that all the library uses MARC 034 fields for storing all their location information. It is surprisingly easy to find reasons to get exasperated by MARC records but they are also a reflection of the technological constraints that people were operating under when they were created. More importantly they worked well enough that, to this day, they continue to work well enough that nothing else has superseded them.

MARC itself has grown to become a large and daunting specification. There are a small handful of general purpose tools for working with MARC records but lacking the time I chose to write a single purpose-fit tool for parsing MARC 034 records in to bounding boxes with decimal coordinates and from there in to GeoJSON **Feature** `https://datatracker.ietf.org/doc/html/rfc7946` records.

My thinking was that to ask the library to change its existing practice, and by extension update its entire catalog retroactively, was both a non-starter and presumptuous. The library had settled on MARC 034. Few, if any, contemporary spatial tools work with MARC 034. What was necessary was something simple to install and easy to use that could bridge those two facts. Which is how the **go-marc** `https://github.com/aaronland/go-marc/` tools came to exist. The tools consist of a software library for parsing MARC 034 records and tools, that use those libraries, for working with documents containing MARC 034 records.

There is a tool for converting arbitrary MARC 034 strings in to decimal bounding boxes. For example:

```
$> ./bin/marc-034 '1#$aa$b22000000$dW1800000$eE1800000$fN0840000$gS0700000'
-70,-180,84,180
```

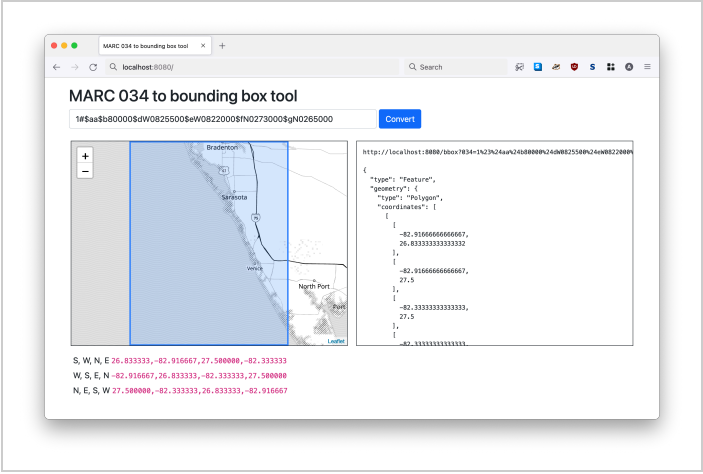
There is a tool for reading one or more CSV files containing MARC 034 column data and appending the corresponding decimal bounding box information to a new CSV file. For example, given a CSV file that looks like this:

```
$> cat test.csv
id,marc_034,name
123,1#$aa$b22000000$dW1800000$eE1800000$fN0840000$gS0700000,example
456,1#$aa$b80000$dW0825500$eW0822000$fN0273000$gN0265000,another example
```

The tool in question would produce the following output:

```
$> ./bin/marc-034-convert -to-stdout ./test.csv
id,marc_034,max_x,max_y,min_x,min_y,name
123,1#$aa$b22000000$dW1800000$eE1800000$fN0840000$gS0700000,180,84,-180,-70,example
456,1#$aa$b80000$dW0825500$eW0822000$fN0273000$gN0265000,-82.3333333333333,27.5,-82.91666
```

Finally there is a tool that launches a simple web application where MARC 034 data can be entered and its bounding box will be displayed on a map. It looks like this:



There's nothing special about the web application. It follows the same pattern that I've been pursuing with similarly-sized map-based applications at SFO Museum. Writing about **a bespoke web application built around the Placeholder geocoder** `https://millsfield.sfomuseum.org/blog/2019/11/04/placeholder/` I said:

> It's a relatively simple web application that depends on three separate JavaScript frameworks, two different services and a whole other cartography component. They are:
>
> 1. ***Placeholder*** `https://github.com/pelias/placeholder` *– The coarse geocoder we've been discussing so far.*

2. ***Bootstrap*** `https://getbootstrap.com/` — *A framework for developing responsive and mobile websites*.

3. ***Leaflet*** `https://leafletjs.com/` — *A framework for web-based mapping applications*.

4. ***Tangram*** `https://github.com/tangrams/tangram` — *A framework for styling and rendering data in web-based mapping applications*.

5. ***Nextzen*** `https://www.nextzen.org/` — *A free and open provider of geographic data for web-based mapping applications*.

6. ***Tangram map styles*** `https://github.com/tangrams?utf8=%E2%9C%93&q=-style` — *The cartography and styles used by Tangram to render data provided by Nextzen. These are also referred to as "scene files"*.

For most of the last decade the conventional wisdom has been to load each of these pieces remotely. Usually these pieces are hosted by some sort of third-party content distribution network (or CDN) to help speed things up and the guts of an application are often little more than the glue that binds all these remote bits of functionality together. It's a fine way of doing things but it's not the way that I wanted to set up a Placeholder service for the museum.

Instead I wanted:

1. *To run a local copy of Placeholder.*

2. *For all of the Javascript dependencies (Bootstrap, Leaflet, Tangram) to be bundled with the application and served locally.*

3. *To be able to read and write cached tiles from Nextzen.*

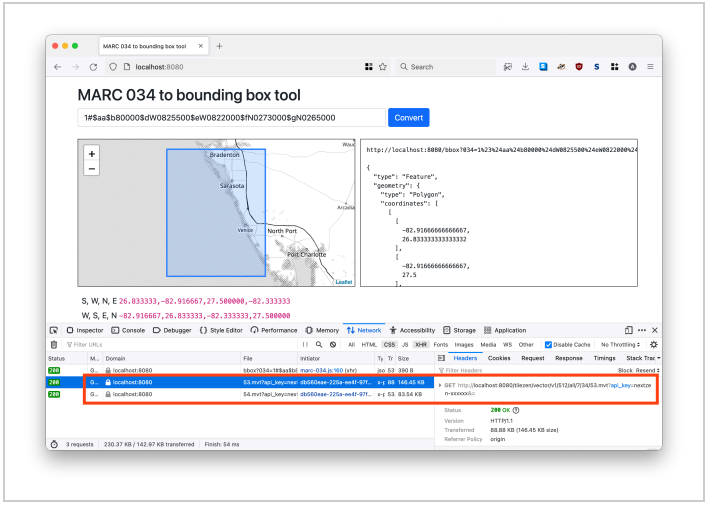
4. *To be able to do all of this both locally, on a remote server and in "the cloud".*

The bounding box converter application does not need to run a local copy of Placeholder but otherwise all the concerns the same. With the bounding box application I've added the ability to use locally bundled copies of the Nextzen geographic data (vector tiles) so that the application can be entirely self-contained and run without any dependencies on third-party services.

These locally bundled data are called "tilepacks"which are just **MBTiles databases** `https://github.com/mapbox/mbtiles-spec` containing vector tile data compiled using the **tilezen/go-tilepacks** `https://github.com/tilezen/go-tilepacks#build` package. To use a local tilepack with the bounding box application pass the path to your database to the -nextzen-tilepack-database flag. For example:

```
$> ./bin/marc-034d -nextzen-tilepack-database tiles/nextzen-world-2019-1-10.db
2021/10/23 14:27:33 listening on http://localhost:8080
```

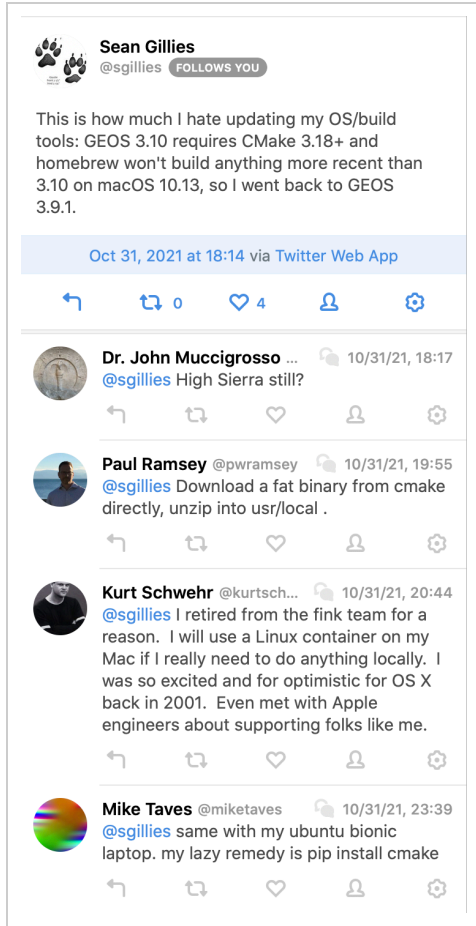
The application won't perform any differently but if you look "under the hood" you'll see that the vector tile data is being served from the application itself.



Locally bundled tilepacks aren't necessarily the solution for every map-based web application. A tilepack with global coverage spanning zoom levels 1 through 10 is a 1.8GB file. Every subsequent zoom level roughly doubles in size. A tilepack with global coverage for zoom level 16, alone, will be over 100GB. Even accounting for optimizations like excluding tiles that are just water the tilepack will be so large that it may start to present a number of operational challenges.

For applications that don't need to account for global coverage, though, or that operate without the need for street-level tiles, this opens us more avenues for simple-yet-sophisticated applications (again, remember the layer-cake of functionality described above) that can be deployed with the guarantee that they'll just keep running independent of third-party services. At SFO Museum I've been working on a similar bundling of map tile data using the **Protomaps toolchain** `https://protomaps.com/` , as well other spatial data, to create a **self-contained geotagging application** `https://millsfield.sfomuseum.org/blog/2021/05/03/geotagging/` .

All of these applications have been written in **Go** https://golang.org/ which, I will be the first to admit, is not always the most pleasant language for developing web applications. There is much about the language that makes writing complex web applications difficult and time consuming. But there is a whole other class of simpler applications, of **small focused tools** https://millsfield.sfomuseum.org/blog/2020/08/04/small-tools/ , where the ability of Go to embed its static assets (its templates, web pages and other resources like CSS or JavaScript files) and in to a single binary application that runs its own web server and can be pre-compiled, inclusive of all its dependencies, for multiple operating systems is where it shines.



It shines because it makes a meaningful effort at removing so much of the operational hassle of other approaches whether it's a steep dependency chain, a steeper learning curve to master the tools to handle that dependency chain, the fickleness or short-term approaches adopted by one or more parts of a dependency chain or all of the above. It's not that other approaches don't have their merits so much as their requirements can be prohibitive.

This is especially true in an environment like the cultural heritage sector which is **lacking the practice of developing a toolkit** https://aaronland.info/weblog/2020/11/12/touching/#mcn2020 of small, simple-to-use and simpler-still-to-install-and-maintain tools. The sector is littered with battleship-sized projects that in their attempt to do everything only succeed in accomplishing nothing.

These tools for parsing MARC 034 records do very little but my hope is that, precisely because they only try to do one thing well, they will allow people to get other things done faster.

These tools also don't solve the actual problem of finding intersecting Who's On First records for bounding boxes. If you are familiar and comfortable with the dependency chain of spatial databases this becomes a trivial problem once your MARC 034 records look and feel like decimal bounding boxes. If not it remains "tomorrow's problem". Importantly it remains a *separate* problem.

https://github.com/aaronland/go-marc/ https://github.com/aaronland/go-marc/

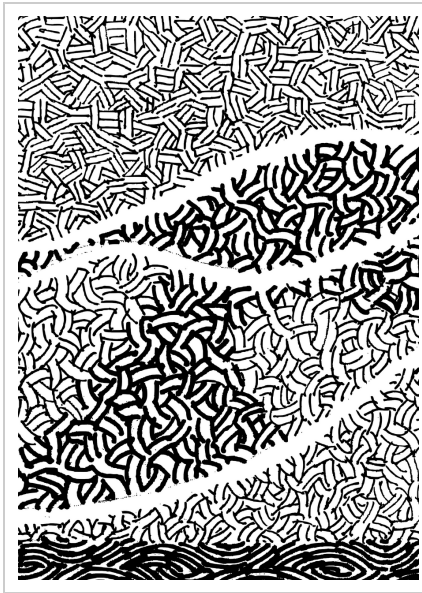
2021-11-01

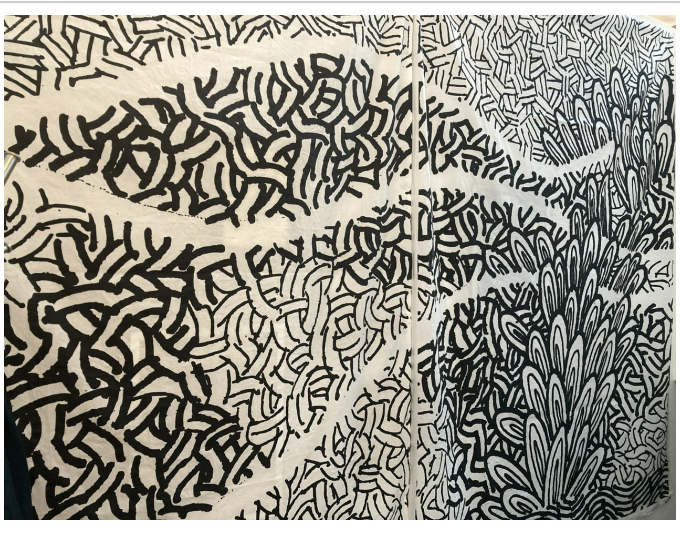
revisiting

Russian River, revisited (2021)

Russian River, revisited (2021)

The original drawing, one of a series of six **from March of this year** `https://aaronland.info/weblog/2021/03/22/oneyear/#views` .





The same, but different. https://aaronland.info/weblog/2020/10/27/uniformity/#same

2021-11-10

continuous partial mythologies

The myth of continuity

The myth of continuity

Part one

Earlier this week, I was part of the ***The Mythologies of Museums*** `https://mcn2021virtual.sched.com/event/lwtk/the`

-mythologies-of-museums *panel at MCN 2021, alongside* ***Susan Chun*** http://susanchun.com/ , ***Sherri Wasserman*** https://spoolfive.com/ *and* ***Bruce Wyman*** http://www.usd-mach.com/ . *The panel was pitched as a "need to understand our shared mythologies about museums, practical examples of how we can do things differently, and the ripple effect across the organization for our most basic decisions." I was asked to speak to this idea in the context of digital technologies in museums. These are the notes I wrote for the panel and used as a kind of opening statement.*

I'd like to talk about the myth of continuity, specifically the myth of achieving continuity in our endeavours. In order to talk about this it is necessary to talk about platforms as vehicles of distribution and of the very important distinction between the web and the larger internet, even though the two are often understood as being the same thing.

There is a reason that companies like Facebook and Snap are talking about virtual, augmented and mixed realities: Simply put they are trying to create and control a new means of distribution that is not a mobile phone. They are doing this because there are exactly two mobile computing platforms and they are the choke point for enabling or disabling features, functionality and revenue.

It is important to understand the web, the thing that Tim Berners-Lee created and released unencumbered of any restrictions in the early 1990s, in this light. The web is simply a set of protocols that sit on top of the internet that define the means by which any two people, one with a server and the other with a client, can exchange documents. That's basically the sum total of the web and that simplicity, coupled with absence of control by any one platform

vendor, is what makes it so important.

The web does not do everything, it is not a magic pony. What it does do, however, is enable two things: One, the ability to project an idea beyond the range of any one individual's voice, beyond the limitations imposed by physical space and other geographies. Two, it allows for the asynchronous recall of those ideas outside the determination of someone else's schedule or editorial bent.

It's not hard to see why this should be important to the cultural heritage sector. One only has to remember, or imagine, the world that preceded the web. It was a world where access to and awareness of our collections was bounded by geography, the constraints of book publishing and the occasional documentary film. The exhibition catalog was **the unit of currency** `https://www.aaronland.info/weblog/2013/05/05/design-thanking/#face` in the cultural heritage sector. A bookshelf of catalogs was the only proof of the arc of an institution's history and work.

In 2021, the myth of continuity suggests this is still largely the case.

The promise of the web, of the internet as a whole but specifically the web because of its effects on the power dynamics of distribution, was not only that we could reach an audience outside of geography but that it could bridge all those exhibition catalogs, and all the materials used to create those catalogs. That bridge would create first a network, a web, of ideas and relationships greater than the sum of its parts and, second, serve as the building blocks for new ways of understanding and interpreting our collections. It would do so not absent of economics but *absent the economics of someone else's*

business model which rarely, if ever, align with the goals of the cultural heritage sector.

That hasn't happened because we are unable or unwilling to believe that we can and should control the means to make these things possible. We are still largely unable to break free of the "fire and forget" model of developing exhibitions and exhibition catalogs. We are largely unable to retain the staff and the institutional knowledge necessary to make any given digital project possible beyond its launch.

Exascerbating this problem is the belief that digital infrastructure is more like a book than, say, a garden. That, within the limits of aging, it can simply sit unattended on a shelf and be revisited only by brushing off any accumulated dust. Just to be clear about something: I love books. Books are amazing. But a field guide of plants is not the same thing as a living, breathing garden and no one pretends that a garden left unattended will flourish.

Exascerbating that problem is the belief that the only way to address these digital and technological short-comings is to adopt the methodologies of the private sector ignoring the simple fact that the private sectors operates at a scale of staffing the cultural heritage sector is unlikely to ever enjoy, at least not in our lifetime. So we compound the problem of staffing, and more specifically of retention, by adopting means and methods designed to tackle a problem we'll never have.

Part two

This is the long-form version of some of the things that I said when the panel moved to an open conversation.

Until recently, the web was still more exciting than not. This fact intersected with the cultural heritage sector's need or desire to be associated with cool, new technologies. It has been a marriage of convenience where, almost by happenstance, the hot new technology (the web) aligned with the values and the mission the sector claims as its own. At the end of 2021, the web is seen by many as "boring" and there is a lot of effort being focused on different forms of **augmented reality environments and immersive experiences** `https://youtu.be/enMwwQy_noI` . It is, some people say, the "future of museums". Perhaps it is, or can be, but there are a couple of issues worth addressing before the cultural heritage sector

begins this new adventure.

The first is that not even Facebook, which is hiring literally tens of thousands of engineers to build these environments, thinks the ideas they are imagining are within reach right now. So not only are they willing to hire large teams dedicated to an ambitious betting-on-the-future project they are committing to sustaining those teams for years to come. Museums and libraries, on the other hand, can still barely keep their own websites running for more than five or ten years before they are rebuilt from scratch as part of whatever new initiative is being advanced by the organization.

In 2018, at a the Conference on Mobile Position Awareness Systems and Solutions, **I talked about this phenomenon** `https://aaronland.info/weblog/2018/09/12/distractions/#diorama` by saying "...that Disney is to the museum sector what Google is to the technology sector."

What I mean to say is that using Disney as a frame of reference when discussing what is possible in the museum space in 2018 is an unhelpful distraction. Saying "But Disney does it..." has become the trickle-down economic theory of technology development in the museum sector.

It also does a real disservice to the genuinely hard work these companies have done and continue to accomplish. Google and Disney exist in universes of their own creation, complete with air that we don't breathe, but it is important to recognize that they have (mostly) earned it.

It is important to recognize that there is a network effect in each of their individual efforts and that those efforts play themselves out over multiple projects often spanning years and many failures. The museum sector, both individually and collectively, remains unable or unwilling to make those kinds of investments or to take equivalent risks and the results are predictably un-Disney-esque.

I wish the museum sector would endeavour to create Disney-scale infrastructures and tooling in service of cultural heritage. Until it does though maybe we can just admit that talking about Disney does nothing to address the needs and concerns we face on a day-to-day basis?

The second issue that we, operating in the not-for-profit or notionally not-entirely-for-profit sector, need to address is whether or not we are responsible for considering the consequence of our participation

in the technologies we choose to embrace. "We go where the eyeballs are", is the excuse we make for engaging on and with platforms whose motivations and behaviours are increasingly difficult to reconcile with anything other than short-term, profit-driven decisions that benefit the few to the cost of the many.

These same platforms are now embarking on efforts to replace the common playing field they were forced to operate on, the web, with environments that they control from end to end. Again, the dilemma that companies like Facebook face being forced to operate within Apple or Google's mobile platforms is instructive. It has always been in Apple's interest to preference native applications over equivalent web-based applications. The former ensures that the device itself is a means of control and so Facebook, being forced to develop and promote native applications, is always subject to someone else's decisions whether they are made out of negligence or malice.

What we are witnessing now is a concerted effort, by a number of different actors, to replace the dominant means of enforcing that control. It remains to be seen whether it will happen any time soon and it remains to be seen whether that control will be used merely to extract new profits or to influence behaviour, assuming their is any difference between those two things anymore.

Absent from all of these discussions is anything like the vision that Tim Berners Lee espoused for the web as an open platform, free of licensing constraints. One that was simple enough to guarantee it could be implemented and operated with limited resources. When we talk about virtual worlds we are not only talking about walled gardens governed by the whim and folly of **shareholder value** `http://www.aaronland.info/weblog/2014/10/06/interpretation/#brick` . We are also talking about infrastructures that

only a few enterprises are in a position to operate.

It is true that cultural heritage institutions *should* go where the people are in order to better raise awareness of and share our collections. At least in principle. At some point, though, we need to ask ourselves whether participating with platforms whose benefits, however real and tangible they may be, are increasingly being dwarfed by their consequences is something that undermines our mission as civic-minded organizations for the common good.

Part three

Almost none of what follows made it in to the panel. We were asked to think about meaningful suggestions of things people might do to address the questions we'd been talking about. What follows is what I would have liked to say if we'd had more time.

Define what "digital" means

Define what digital means for *your* organization. My own definition of digital is expansive and that is why I put so much weight on the value and importance of the web. My definition doesn't need to be your definition but we owe it to ourselves, and to our colleagues, to articulate what we mean when we talk about "digital". For too long important differences in that understanding have been **lumped together under the same umbrella term** `https://aaronland.info/weblog/2020/04/06/futures/#capacity` and it is one of the reasons the sector has difficulty implementing any of its goals.

Make the web your baseline

Make the web your baseline, in particular a web of documents and links rather than one of interactions or transactions or experiences. This is the profoundly boring web but that is the point. It is the web that, structurally, has a better guarantee of outlasting the machinations of third-party vendors. In the time between the panel at MCN and this blog post Google has updated its Chrome browser engine to **make it possible to prevent viewing the source of any given web page** `https://chromium-review.googlesource.com/c/chromium/src/+/3260807` violating one of the core principles, what Tim Bray has called **the View Source lesson** `https://www.tbray.org/ongoing/When/200x/2003/05/21/RDFNet` , of the web.

> Adding the ability to block View Source is antithetical to what the web used to stand for. It is profoundly fucking evil, and everyone responsible should be ashamed. I don't care how many times you say the word "enterprise" as an excuse for your decision.
>
> —
>
> **jwz** https://www.jwz.org/blog/2021/11/putting-on-my-hacker-hoodie-to-view-source/

I have joked nervously that one day we would all have to start compiling our own browsers from source again to account for nonsense like Google's changes. I have always hoped I was being paranoid. Maybe, maybe not. It does serve to make my point that we should remember that what the made the web different, what distinguished it from the past, was not a lot of fancy interactivity but the ability for common resources (text and images, even video) to hold hands across the internet using a model so simple that it could be built with only modest effort.

No one can pretend that web browsers are "simple" anymore, but that's not really my point. Neither is it that we should do away with all the complex and sophisticated interactivity we've developed over the years. I am not suggesting that we return to the web of the mid-1990s with its grey backgrounds and blue and purple links. I am suggesting however that we ensure that web is our guaranteed *failure scenario*, should it ever be necessary, and that we layer everything on top of it.

This is as much about safe-guarding against other people's browser decisions as it is giving the cultural heritage sector the freedom to engage with the walled gardens of platform vendors, without that participation turning in to a black hole from which it is impossible to escape.

I am suggesting that we evaluate all of our technological choices through a lens that forces us to unravel and understand layers and layers of abstractions and conveniences so that we preserve the technological, conceptual and economic freedom that the web made possible.

Publish data first

Publish data first, and then build on top of it. I spoke about this, at Museums and the Web 2019, in the context of **the work SFO Museum is doing with its websites** `https://www.aaronland.info/weblog/2019/04/08/post/#mw19` :

Historically the model for most digital or web-based initiatives has been to first export data from an internal collections management system. Second, that data is massaged in to an intermediate form for use by the project at hand and then third, exported again in to a typically bespoke machine-readable format.

We have changed the order of things to publish the open data representation first and then, from there, to build our own websites and services on top of that.

Everything I've described so far has been built using the same raw materials that we've made available for you to do something with. This introduces a non-zero cost in the build process for the public-facing museum efforts but we believe it's worth the cost.

First of all we want other people to build new interfaces and new services, new "experiences" even, on top of our collection so this is a way to keep ourselves honest. If we can't build something with this stuff why should we imagine you will?

Second, we want to ensure that the data we release and the manner in which it is published, is actually robust and flexible enough to engender a variety of interfaces and uses because we need that variety. It is important to the museum because I don't believe there is, or should be, only one master narrative in to the collection.

In many ways this is just the actual implementation of everything I

described in the last section. This is the boring web of documents and links. It is the boring web that makes all the other experiences and interactions possible.

Small focused tools.

Make the extra effort, whenever possible, to extract those pieces in the bespoke tools written by and for an organization in to smaller pieces that can be shared with others. It's hard to overstate how important this is for the cultural heritage sector, at least in the near-term. The sector is still a long ways from finding any kind of meaningful solutions to **the problems of first hiring and then retaining the staff** `https://aaronland.info/weblog/2020/11/12/touching/#mcn2020` who can build and maintain its digital systems. Until that problem is resolved every institution is going to be operating with too few people, doing too many different things. They will be developing infrastructures and scaffoldings that are purpose-fit to the realities of their situation, often cutting corners out of necessity.

None of these things can or will be designed from first principles or a conceptual framework that supports anything but the institution where they were built. The staffing realities of the cultural heritage sector dictate that simple, unfortunate reality. That doesn't mean we can't or shouldn't share resources, though. It simply means we need to think differently about how and what we share. We need to develop the practice of thinking about our tools as layers of components where those components that can be shared are simple enough for someone at a different instituion, someone who is just as busy and distracted as you are, to quickly understand what something does or doesn't do.

The emphasis here is on "quickly" because the reality is that other people will be looking at whatever you've shared over lunch or on the train going home. The emphasis needs to be on quickly because the ultimate goal is to have something that can be understood with sufficient ease that it can be filed away to be remembered *at some later date* when there is a need to address a specific problem. If the goal is to build a common "kit of parts" that can be re-used across the cultural heritage sector the first step is to make awareness of those tools, and their conceptual boundaries, a practical and tangible reality.

I've been trying to do this, **and writing about it** `https://millsfield.sfomuseum.org/blog/tags/tools/` , with **the toolchain** `https://github.com/sfomuseum/` we're building at SFO Museum.

Write about it.

Write about it, whatever "it" is, and publish it on the web for the future to find. In the same way that we need small focused tools the sector needs to tell itself the stories, good and bad, of its work. The sector needs to demonstrate to itself, and to its broader audiences, that there is proof of life in its work.

Be generous with your thoughts have confidence that someone will find value in your work, whatever it is. Share your experience with the understanding that the value in what you are doing is in the act of sharing itself. The measure need not be "attracting eyeballs" but in participating and giving back to the common resource that the web can be.

If you have ever benefited from something someone else has shared

online it stands to reason that someone else, in turn, will benefit from what you have to share. Weeks or months, years even, may pass before someone else finds what you've written but it's important to understand: This is what separates the web from what came before it.

The web has made it *possible*, both technologically and economically, to allow thoughts and ideas to survive the long-term. The web is the means by which an idea outlasts the reluctance of the present. It is one way that we might realize the myth of continuity.

2021-11-12

things I have written about elsewhere #20211119

Interactive maps of SFO from 1930 to 2021 (and...

Interactive maps of SFO from 1930 to 2021 (and beyond) at the T2 SkyTerrace

This was **originally published on the Mills Field weblog** https://millsfield.sfomuseum.org/blog/2021/11/19/skyterrace/ , in November, 2021.

The recently opened **Mills Field and the San Francisco Airport** https://www.sfomuseum.org/exhibitions/mills-field-and-san-francisco-airport exhibition in the **SkyTerrace observation deck in Terminal 2** https://www.flysfo.com/skyterrace includes an interactive map installation that allows visitors to view the history of SFO by browsing over three dozen aerial maps from 1930 to 2021. Orginally designed as a touch-based interactive application, designed to work entirely offline, it was updated earlier this year to allow visitors to control the map using their personal mobile devices. The installation displays a QR code that a visitor can scan which will launch a

"controller" application that relays instructions from their mobile device to the application running in the SkyTerrace. After 5 minutes a new QR code will be displayed allowing another visitor to browse the map collection.

All of this builds on work the Museum has been doing, since 2018, to make these maps available online on the **Mills Field website** `https://millsfield.sfomuseum.org/map/` . These maps reflect the growth and changes at the airport over the years and decades and you can see them online at: **https://millsfield.sfomuseum.org/map/** `https://millsfield.sfomuseum.org/map/`

There are maps available from 1930 to 2021 drawing on the **Museum's own collection** `https://millsfield.sfomuseum.org/search/?q=aerial+photograph` , the **Special Collections Library at the University of California Santa Baraba** `https://www.library.ucsb.edu/special-collections` and the airport's GIS department.

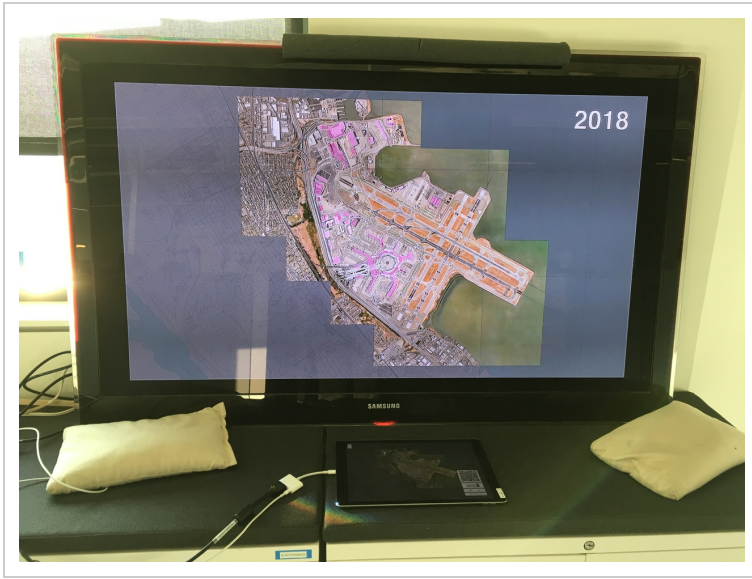
Our goal is to have aerial map imagery for every year since the airport's inception. We haven't quite gotten there yet but there is at least one map for every decade since the airport opened (and year-over-year maps since 2012).

The interactive map allows visitors to change the viewport to see the airport relative to the greater Bay Area and to hide or show the present-day airport's footprint to better understand the scale and velocity of changes that have been made to the airport campus.

Every part of the SFO terminal complex, with the exception of the pre-security common area in Terminal 2, has changed since 1954. In 1997 there were cars driving through what is now the International Terminal.

The more recent maps, from 2015 onwards, are so detailed that you can get up close to individual planes as they taxi out to the runways.

In 2020, because of COVID-19, there was a little less taxi-ing than usual.



This is what the first version of the SkyTerrace interactive looked like. As mentioned the installation was originally designed as a touch-based interactive using an iPad attached to an external monitor. Under the hood it's a simple web application wrapped in a native iOS application. In **a blog post I wrote about making some of that work available as open-source software** `https://millsfield.sfomuseum.org/blog/2020/11/18/multiscreen/` I said:

> Our goal is to treat this as a fast, cheap and reusable template for building interactive installations. I am fond of telling people that we have developed a system for interactive displays with minimal hardware and stable costs, that work both online and offline, and that we can deploy to any four-by-four foot space at the airport. The only question should be: What do we want the next interactive to do?

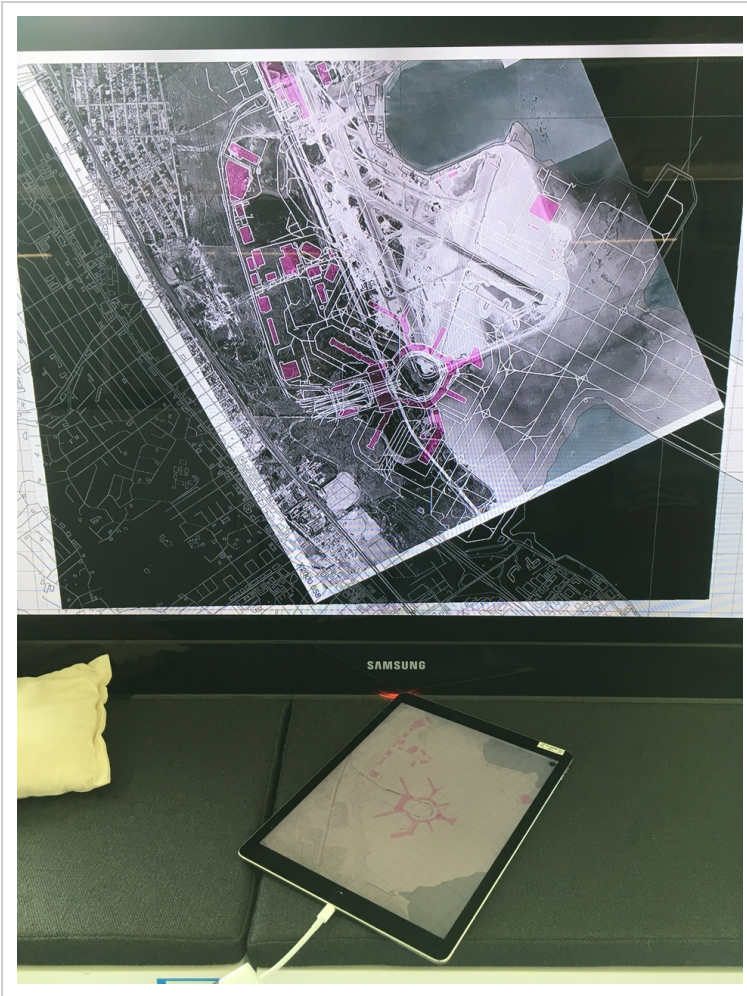
We haven't yet made any of the tools that allow the map to be controlled from a visitor's mobile device available as open-source software but we hope to soon.

What follows are images and screenshots documenting the earliest

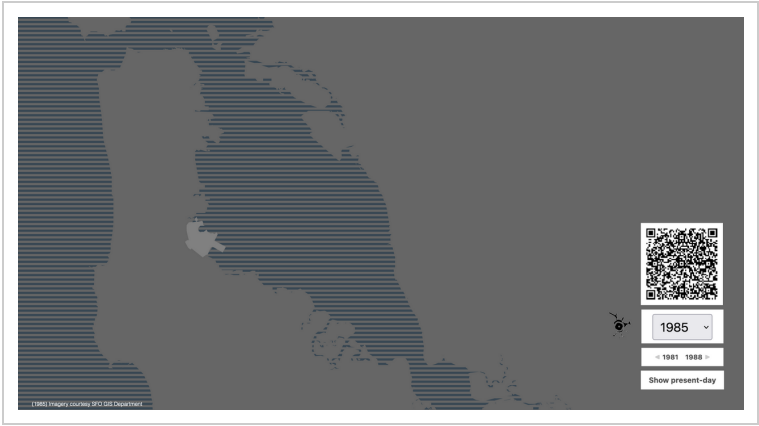
prototypes to the installation now on view in Terminal 2. This is the very first iteration of the application showing some early ideas for design elements. In these early versions of the application the QR code would link to the same **map on the Mills Field website** `https://millsfield.sfomuseum.org/map/` allowing a visitor to share a link to the map they were browsing in the SkyTerrace.

In this version you can see the outlines of the final interface. You can also see that the display on the iPad is simply being mirrored to the external display. The decision to prevent this from happening, to have a single application with multiple screens each showing different content, is one reason we would eventually settle on developing a native iOS application that wraps a simple web

application.

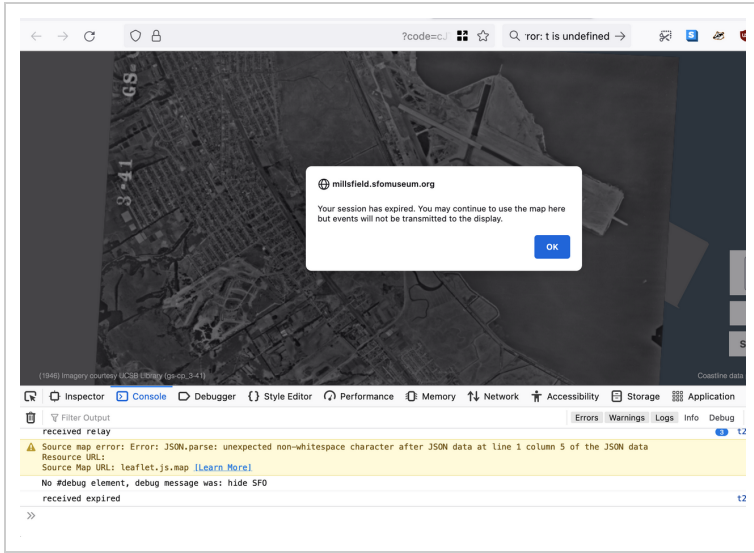


The first versions of the application also used a very detailed, but ultimately distracting, cartography to depict the present day airport and surrounding Bay Area.

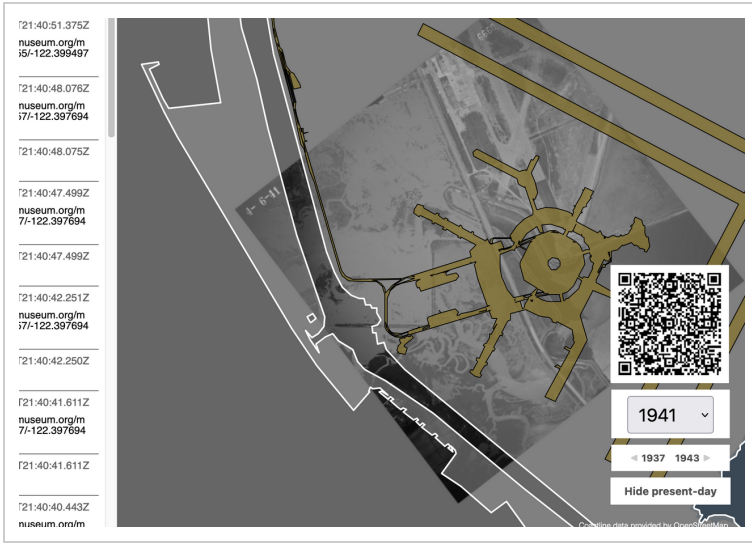


Once the decision was made to refactor the touch-based application in to something could be controlled remotely I also took the opportunity to revisit the cartographic elements that frame the historical maps. To start, the detailed road networks of the San Francisco Bay Area were de-emphasized in favour of the Bay itself.

A similar approach was taken with the airport campus itself focusing only on those features of the airport that passengers are likely to interact with when flying in and out of SFO. We are grateful for the help and assistance that the airport's GIS department provided in making these data available enabling us to develop systems that will track changes to the airport's physical footprint over time.

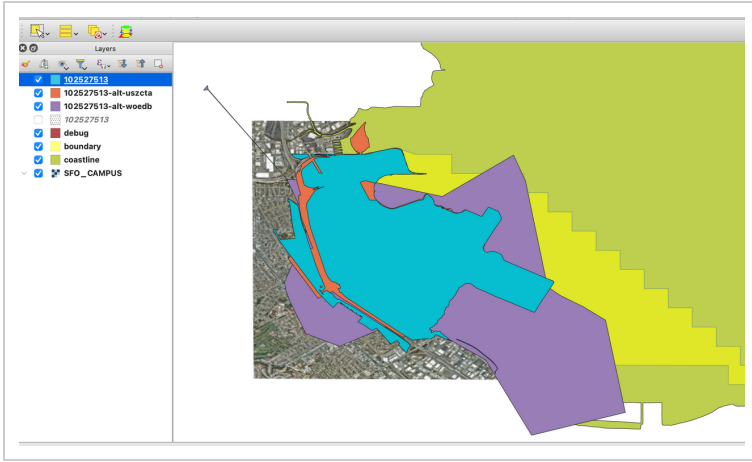


The switch to a network-enabled application with multiple users necessitated a lot of juggling of different states and ensuring relevant feedback for different scenarios. Here's an example of a message that a passenger would see if their interactive session has expired and someone else's device is now controlling the map.

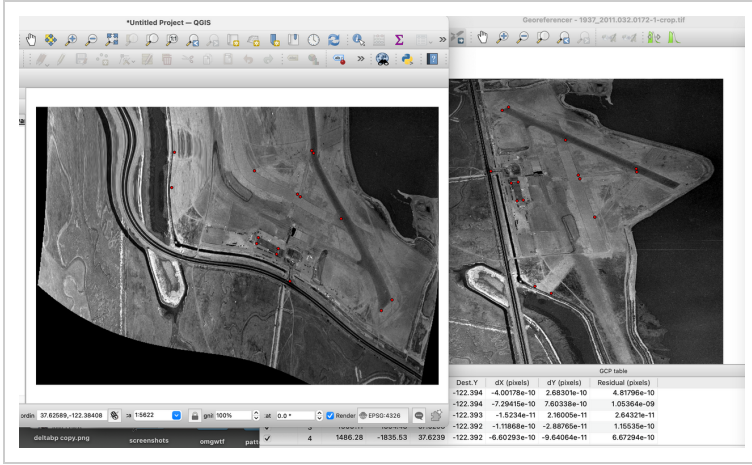


This image shows the design and interaction elements starting to settle in to what's on display today, as well as a hint of the volume of debugging information the application produces off to the side.

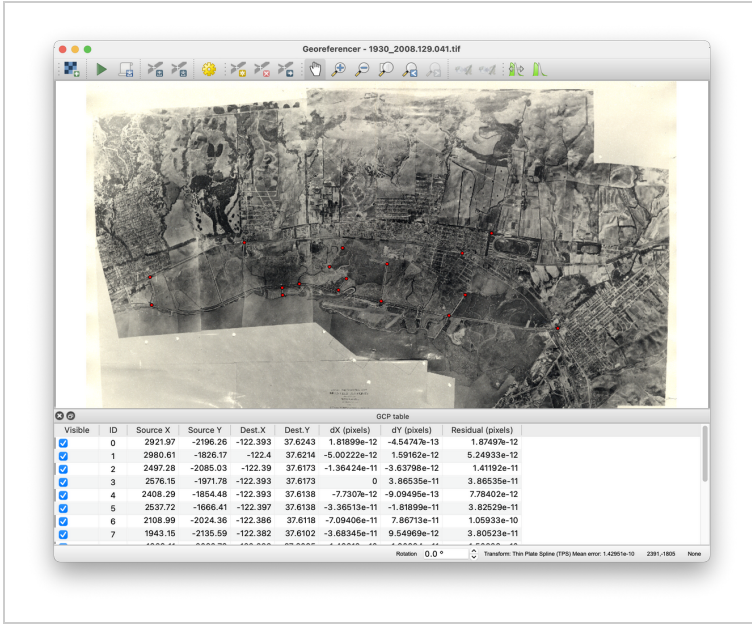
This is a screenshot of the mobile "controller" application which is designed to mirror, as much as possible, the interactive itself. Here you can see that stripes used to represent the San Francisco Bay have been replaced by a solid colour and some outstanding layout issues for creditlines displayed on a mobile device.



The geometry for the San Francisco Bay coastline was sourced from the **OpenStreetMap** `https://www.openstreetmap.org/` project and required a little extra GIS processing to line up with the airport's own geometry for the airport campus.



Many of the aerial maps needed to be "georectified" to match the **map projection** `https://en.wikipedia.org/wiki/Map_projection` used by web-based applications. This is done by making a table of latitude and longitude coordinates in a target projection that correspond with x and y pixel coordinates in a source image. These are the red dots in the image above. Once that mapping has been done it can be used to warp the source image. It doesn't always work especially for images taken at an angle, as you can see here.



Here's another example of aerial imagery from 1930 where the North-South axis of the airport is displayed right to left rather than top to bottom. Because the photographs were taken from an airplane flying directly overhead the warping operation is much more

successful.

This is what it looks like, after being warped, with aerial imagery from 2020 overlayed. **This map** `https://millsfield.sfomuseum.org/map/#1930/14/37.6160/-122.3847` is now on view in the SkyTerrace.

The installation will be continuously updated as new maps become available. Earlier this month we added aerial imagery of the airport from **August, 2021** `https://millsfield.sfomuseum.org/map/#2021/14/37.6160/-122.3847` and if you look closely you can see the repairs to runway 28R underway. The **SkyTerrace observation deck in Terminal 2** `https://www.flysfo.com/skyterrace` is open to the general public from 10AM to 6PM Thursday through Friday and we hope that you are able to stop by for a visit!

As mentioned all of the work for the Sky Terrace interactive builds on a work the Museum has been doing around historical maps since 2018. If you'd like to read more these blog posts (sometimes highly technical) describe some of that work:

1. ***Map updates, 2019 -***

2020 https://millsfield.sfomuseum.org/blog/2020/01/10/map/ , *January 2020*

2. ***More recent old maps (and the shapes in the details)*** https://millsfield.sfomuseum.org/blog/2019/11/06/maps/ , *November 2019*

3. ***More old maps and more-better architectures*** https://millsfield.sfomuseum.org/blog/2018/10/04/architecture/ , *October 2018*

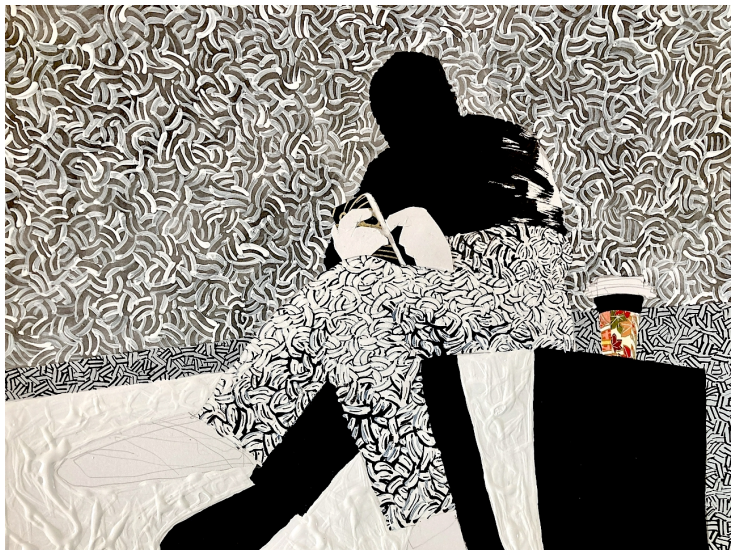
4. ***Old maps (and old map tiles) at SFO Museum*** https://millsfield.sfomuseum.org/blog/2018/08/07/old-maps/ , *August 2018*

2021-11-19

deep-dreaming in letraset

Six works with chiyogami (November, 2021)

Six works with chiyogami (November, 2021)



These works are **available for sale** `https://or-this.myshopify.com/collections/chiyogami` .

2021-11-22

things I have written about elsewhere #20211230

The Accession Number Project

The Accession Number Project

This was **originally published on the SFO Museum Mills Field weblog** `https://millsfield.sfomuseum.org/blog/2021/12/30/accession-numbers/` , in December 2021.

Boarding pass: Air France, Concorde. Paper, ink. Gift of G. Robert "Bob" Hamrdla, SFO Museum Collection. ***2011.066.057***

The goal of the "Accession Numbers" project is to compile a catalog of machine-readable patterns for identifying and extracting accession numbers in arbitrary bodies of text for as many museums and cultural heritage organizations as possible. The project is a continuation of some experimental work, started by the **Cooper Hewitt** `https://www.cooperhewitt.org` in 2014 to use optical character recognition (OCR) software to extract accession numbers from photographs of wall labels. Here's a quote from **a blog post describing that work** `https://labs.cooperhewitt.org/2014/label-whisperer/` :

> Conveniently, accession numbers are so unlike any other element on a wall label as to be almost instantly recognizable. If we can piggy-back on Tesseract to do the hard work of converting pixels in to words then it's pretty easy to write custom code to look at that text and extract things that look like accession numbers. And the thing about an accession number is that it's the identifier for the thing a person is looking at in the museum.

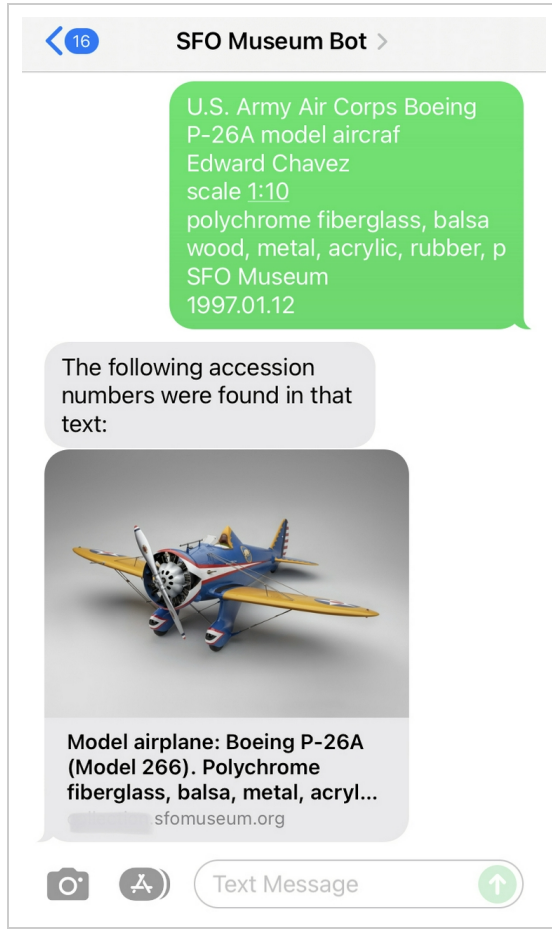
Full disclosure: I wrote that blog post. Although all the work in that post was

focused on wall labels at Cooper Hewitt we imagined it was an approach that could be extended easily enough to other museums. Due to time and **other circumstances** `http://mw2015.museumsandtheweb.com/paper/strategies-against-architecture-interactive-media-and-transformative-technology-at-cooper-hewitt/` it was an idea that germinated but then remained dormant until this year after Apple announced the introduction of **Live Text** `https://support.apple.com/en-us/HT212630` in iOS 15. What features like Live Text, and Google's **Lens** `https://lens.google/` product before it, take care of is handling all the hard work of performing OCR operations on a photograph even, and especially, when that photograph was taken at a funny angle or in poor lighting conditions.

The operating system can take as its input a photograph and return all the text that it finds in that image which is pretty exciting in and of itself but becomes especially interesting when we observe that:

> ...accession numbers are so unlike any other element on a wall label as to be almost instantly recognizable.

The reason this is interesting is that people take photographs of wall labels in museums, usually with their phones. If that phone's operating system is capable of extracting text, without the need for a person to install any additional software, then it opens up a whole world of avenues for taking those pictures of wall labels and connecting them to their online counterparts of the objects they describe.



Importantly, nearly all of these avenues are open for any museum to pursue and there is value and benefit in developing tools and applications that can be adopted, and adapted, by as many organizations as possible. Even if SFO Museum chose to approach these opportunities selfishly there is value in ensuring that anything we build, for ourselves, also works in other museums. Everyone who passes through SFO and SFO Museum is coming from or going to *somewhere else*. There are a lot of museums in all those somewhere elses and it would be pretty great if anything we developed for reading wall labels in our galleries worked in the galleries of the place where visitors are traveling to.

We've been investigating some of those avenues and they are discussed in detail

below.

accession-numbers

Boarding pass: Royal Nepal Airlines. Paper, ink. Anonymous gift, SFO Museum Collection. ***2010.013.002***.

If we are trying to develop tools that are not specific to any one organization the first piece in the puzzle is a catalog of machine-readable patterns for identifying and extracting accession numbers. That's what the `sfomuseum/accession-numbers` repository is. It doesn't provide any functionality but aims to be a common reference for accession numbers that any application can use in an automated fashion.

It is organized as a collection of JSON-encoded files whose names match the domain of the institution they describe. The simplest defintion files contain an `organization_name` and `organization_url` property as well as one or more patterns (and tests to validate each pattern) stored in a `patterns` array. This approach tries to reflect the reality that no two organizations will have the same

needs or concerns in how they choose accession numbers and that trying to derive a universal pattern applicable to all wall labels is almost guaranteed to end in failure.

Here is an abbreviated version of **rijksmuseum.nl.json** `https://github.com/sfomuseum/accession-numbers/blob/main/data/rijksmuseum.nl.org.json` data file for the **Rijksmuseum** `https://www.rijksmuseum.nl` in Amsterdam:


```
{
    "organization_name": "Rijksmuseum",
    "organization_url": "https://www.rijksmuseum.nl",
    "object_url": "https://www.rijksmuseum.nl/en/collection/{accession_number}",
    "whosonfirst_id": -1,
    "patterns": [
    {
            "label": "common",
            "pattern": "((?:\\-?(?:[A-Z]{1,}|(?:\\d[A-Z]{1,}))\\-[0-9A-Z]+(?:(?:\\-[0-9A-Z]+){1,})?)
            "tests": {
                "AK-BR-325": [
                    "AK-BR-325"
                ],
                "2RP-F-2001-7-1020-41": [
                    "2RP-F-2001-7-1020-41"
                ],
                "Title(s) Ovoid vase with a red glaze\\nObject type vase\\nObject number AK-BR-325\\
                    "AK-BR-325"
                ],
                "This is an object\\nGift of Important Donor\\nAK-NM-13198-C\\n\\nThis is another ob
                    "AK-NM-13198-C",
                    "V-570",
                    "2RP-F-2001-7-1020-41"
                ]
            }
        }
    ]
}
```


As mentioned, the **schema for definition files** `https://github.com/sfomuseum/accession-numbers/blob/main/schema/definition.schema.json` requires that all records contain `organization_name`, `organization_url` and `patterns` properties.

More complex definition files may include one or more properties containing URI Templates (**RFC 6570** `https://datatracker.ietf.org/doc/html/rfc6570`) that can be used to derive URLs referencing detailed information about an object. These might be an `object_url` property for deriving the web page for an object on an institution's website or `oembed_profile` or `iiif_manifest` properties for retrieving a machine-readable **oEmbed profile** `https://oembed.com/` or **IIIF Manifest** `https://iiif.io/get-started/` for that object.

Definition files may also include an optional `whosonfirst_id` property pointing

to the **Who's On First** `https://whosonfirst.org` record, and geographical information, for that institution. These data could then be used to automatically determine which definition file to use when extracting accession numbers from a text based on a person's location.

1. ***https://github.com/sfomuseum/accession-numbers*** `https://github.com/sfomuseum/accession-numbers`

go-accession-numbers

Boarding pass and ticket jacket: TWA (Trans World Airlines). Paper, ink. Gift of Jan Boelke, SFO Museum Collection. ***2002.029.013***

This is a **Go language package** `https://golang.org` package for parsing `accession-number` definition files and using them to extract identifiers from text. For example:

```
package main

import (
```

```
        "encoding/json"
        "fmt"
        "github.com/sfomuseum/go-accession-numbers"
        "os"
)

func main() {

        var def *Definition

        r, _ := os.Open("fixtures/sfomuseum.json")

        dec := json.NewDecoder(r)
        dec.Decode(&def)

        texts := []string{
                "2000.058.1185 a c",
                "This is an object\nGift of Important Donor\n1994.18.175\n\nThis is another object\n
        }

        for _, t := range texts {

                matches, _ := accessionnumbers.ExtractFromText(t, def)

                for _, m := range matchess {
                        fmt.Printf("%s (%s)\n", m.AccessionNumber, m.OrganizationURL)
                }
        }
```

This would yield:

```
2000.058.1185 a c (https://sfomuseum.org/)
1994.18.175 (https://sfomuseum.org/)
1994.18.165 (https://sfomuseum.org/)
1994.18.199a (https://sfomuseum.org/)
2000.058.1185 a c (https://sfomuseum.org/)
```

This package is used by the `twilio-handler` application, used to make accession number extraction available via SMS messaging, described below.

1. ***https://github.com/sfomuseum/go-accession-numbers*** `https://github.com/sfomuseum/go-accession-numbers`

swift-accession-numbers

https://collection.sfomuseum.org/objects/1511921127/

Boarding pass: Pan American World Airways. Paper, ink. Gift of Barnaby Conrad III, SFO Museum Collection.
2001.038.150 https://collection.sfomuseum.org/objects/1511921127/ .

This is a **Swift language package** `https://swift.org/` package for parsing `accession-number` definition files and using them to extract identifiers from text. For example:

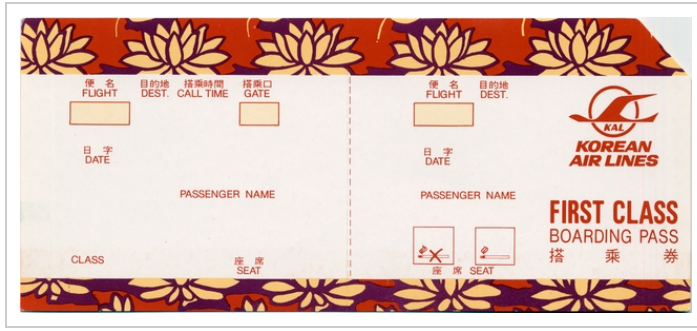
```
import Foundation
import AccessionNumbers

let decoder = JSONDecoder()

guard let url = URL(string: "file:///usr/local/data/sfomuseum.org.json") else {
        // error handling goes here
}

var data: Data
var def: Definition

do {
        data = try Data(contentsOf: url)
} catch (let error) {
        // error handling goes here
}

do {
        def = try decoder.decode(Definition.self, from: data)
} catch (let error) {
        // error handling goes here
}

let candidates  = [Definition]( def )
let text = "This is an object\nGift of Important Donor\n1994.18.175\n\nThis is another objec

let rsp = ExtractFromText(text: text, definitions: candidates)

switch rsp {
case .failure(let error):
        // error handling goes here
case .success(let matches):

        for m in matches {
                print("\(m.accession_number) (\(m.organization))")
        }
}
```

This package is used by the `ios-label-whisperer` application, used to make

accession number extraction available as an iOS application, described below.

1. ***https://github.com/sfomuseum/swift-accession-numbers*** `https://github.com/sfomuseum/swift-accession-numbers`

twilio-handler



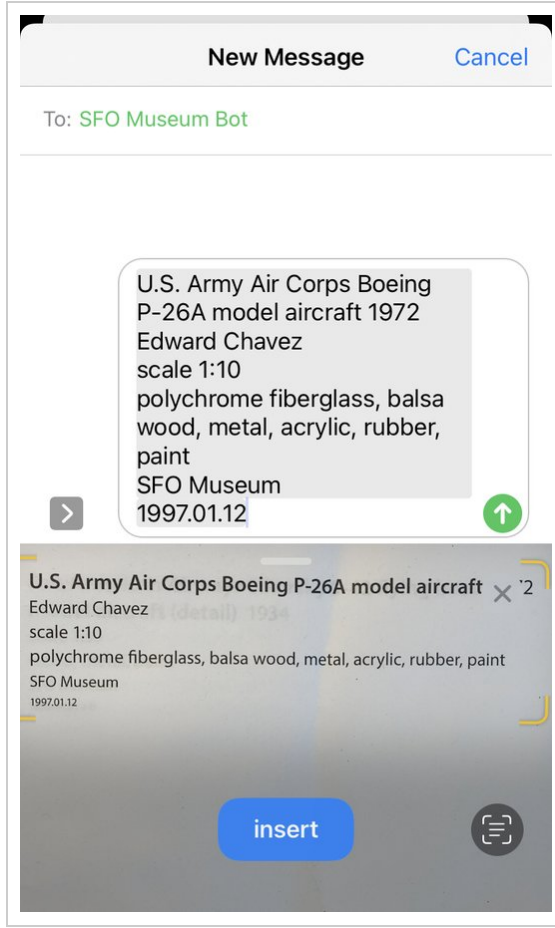
Boarding pass: Korean Air Lines. Paper, ink. Gift of the Frank J. Lichtanski Collection, SFO Museum Collection. ***2011.061.0324***.

The `twilio-handler` application provides an HTTP server to listen for and respond to messages, containing text with accession numbers to extract, sent via SMS to **Twilio-operated** `https://twilio.com` phone-number-as-a-service endpoints. When you create a phone number with Twilio people can send that number an SMS message and Twilio will forward the message to an HTTP endpoint, in this case the `twilio-handler` server, and that then returns the response as an SMS reply.

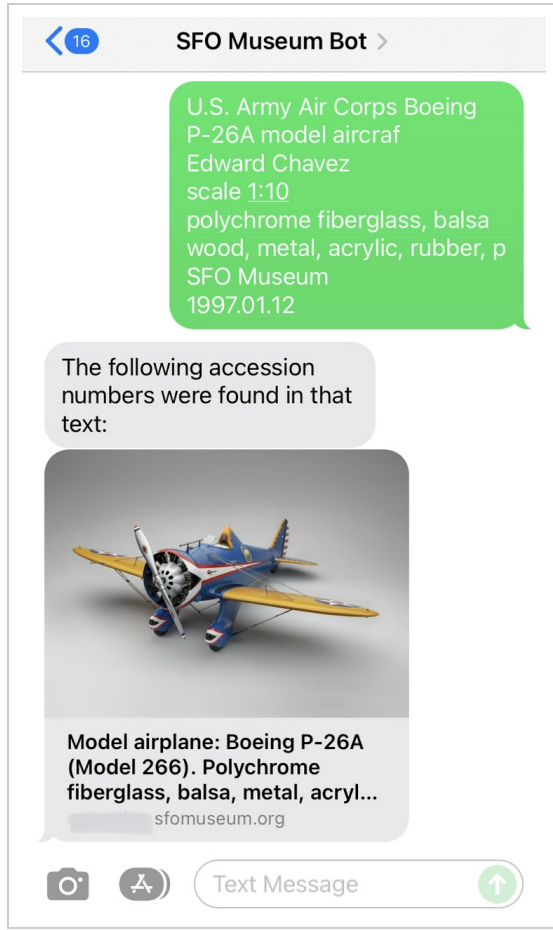
This server can be run locally or using the AWS Lambda + API Gateway pattern. For example this is how it would work locally sending it a Twilio-style message:

```
$> ./bin/twilio-handler -definition-uri 'file:///usr/local/sfomuseum/accession-numbers/data/sfomuseu
2021/12/20 12:01:11 Listening on http://localhost:8080

$> curl -X POST -H 'Content-type: application/x-www-form-urlencoded' -d 'Body=Hello world 1994.18.17
The following objects were found in that text:
https://collection.sfomuseum.org/objects/1994.18.175
```

Here's what that looks like with a real phone number when combined with the iOS Live Text functionality integrated in to the messaging application:



When you scan a wall label with your camera the operating system extracts the text from that image and inserts it as the body of your message.



When you send the message to the phone number operated by Twilio it relays the message to an instance of the `twilio-handler` server which extracts all the accession numbers it finds and then returns a list of object URLs (using the `object_url` URI template described above). That response is then sent back to your phone. Because the message contains URLs the operating system tries to "unfurl" or "expand" each URL in to a short preview.

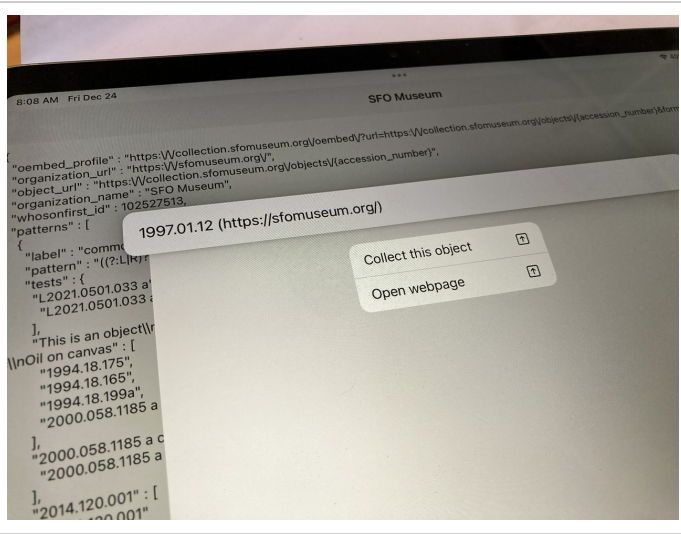
The `twilio-handler` application is designed to work with any definition file in the `accession-numbers` repository so long as it contains an `object_url` URI template.

1. ***https://github.com/sfomuseum/go-accession-numbers#twilio-handler*** `https://github.com/sfomuseum/go-accession-numbers#twilio-handler`

ios-label-whisperer

Boarding pass: Air China. Paper, ink. Gift of the William Hough Collection, SFO Museum Collection. ***2009.108.164***

The `Label Whisperer` is an experimental iOS prototype application that allows a person to scan wall labels from any organization cataloged in the `accession-numbers` repository. Scanned accession numbers can be "collected" to an on-device database and, depending on the functionality described in that organization's definition file, used to load object web pages.



So far, all of the work on `Label Whisperer` has been focused on the underlying "plumbing" rather than design or user experience so it's very rough around the edges still. As with most iOS applications there are a lot of details to account for and this remains a "margins of the day" project so things don't always happen quickly. The goal is not specifically to build an iOS application that SFO Museum publishes (but maybe!) so much as to understand what is possible and importantly to prove that the definition files in the `accession-numbers` repository *can* be used to build an iOS application. Here's a short video showing everything I've just described.

In an ideal world, the `Label Whisperer` application is something that a group of museums, and cultural heritage organizations, could collaborate on and maintain making available to *all of our visitors* as a common platform for collecting objects across *all of our collections*. If that's interesting to you please **get in touch!** `https://github.com/sfomuseum/ios-label-whisperer/issues`

1. ***https://github.com/sfomuseum/ios-label-whisperer*** `https://github.com/sfomuseum/ios-label-whisperer`

"Fake Accession Number APIs"

Boarding pass: Pan American World Airways. Paper, ink. Gift of David A. Abercrombie in memory of Stanley A. Abercrombie, SFO Museum Collection. ***2001.039.433***

One of the things this work has revealed is that lots of museums don't allow individual object web pages to be accessed using an accession number. You can usually specify an accession number as part of a general search and the query will return the object in question but it's not possible to link directly to that object (using an accession number). Many cultural heritage organizations publish their collections as open data and those data usually contain both an object's accession number and the ID associated with that object's web page. With that in mind, I've been working on a side-project called "Fake Accession Number APIs" to create tools which use those open data resources to index, query and resolve accession numbers to their online counterparts.

For example, here's how you would index the open data published by the **National Gallery of Art** `https://github.com/NationalGalleryOfArt/opendata` (NGA):

```
$> bin/import \
        -database-uri 'sql://sqlite3?dsn=accessionumbers.db' \
        -source-uri nga:// \
```

```
        /usr/local/data/nga/opendata/data/objects.csv

# Time passes...

$> sqlite3 accessionnumbers.db
SQLite version 3.36.0 2021-06-18 18:58:49
Enter ".help" for usage hints.
sqlite> SELECT COUNT(object_id) FROM accession_numbers;
136612
```

This is how you would look up a given accession number from the NGA collection using the index you've just created:

```
$> bin/lookup \
        -database-uri 'sql://sqlite3?dsn=accessionnumbers.db' \
        -source-uri nga:// \
        1994.59.10

89682
```

There is also a tool that provides an HTTP server (run locally or using the AWS Lambda + API Gateway pattern) which can resolve and redirect an accession number to its online resource.

```
$> ./bin/server -database-uri 'sql://sqlite3?dsn==accessionnumbers.db'
2021/11/30 12:20:36 Listening for requests on http://localhost:8080

$> curl -s -I 'http://localhost:8080/redirect/?source-uri=nga://&accession-number=1994.59.10'

HTTP/1.1 302 Found
Content-Type: text/html; charset=utf-8
Location: https://www.nga.gov/collection/art-object-page.89682.html
Date: Mon, 20 Dec 2021 06:56:23 GMT
```

The `fake-accession-number-apis` project shouldn't need to exist. However, until the day when it is no longer necessary it can serve as a low-cost and low-maintenance bridge enabling a collection's online resources to work with tools that are derived from the `accession-numbers` project which can't know about online resource identifiers since they aren't printed on wall labels.

1. ***https://github.com/aaronland/fake-accession-number-apis*** `https://github.com/aaronland/fake-accession-number-apis`

Photograph: San Francisco International Airport (SFO), United Airlines. Photograph. SFO Museum Collection. ***2011.068.121***

The ability for our visitors to connect their in-gallery experience of our collection objects with their online and digital counterparts using simple, readily available and familiar tools like their camera phone or their messaging application feels like an important step forward for the sector. All of these efforts are works in progress and we are sharing them now in a spirit of generosity. We welcome your comments and suggestions for how to improve things. If your museum or institution isn't already cataloged in the **accession-numbers** `https://github.com/sfomuseum/accession-numbers` repository we would love for you to tell us about it.

A special thanks to ***Bruce Wyman*** `http://www.usd-mach.com/` *who has so far done the lion's share of the work to compile patterns and tests for the catalog of museums and cultural heritage institutions in* ***accession-numbers*** `https://github.com/sfomuseum/accession-numbers` *repository.*

2021-12-30

internet protocol over rhetorical questions

December 31, 2021

December 31, 2021

2021-12-31